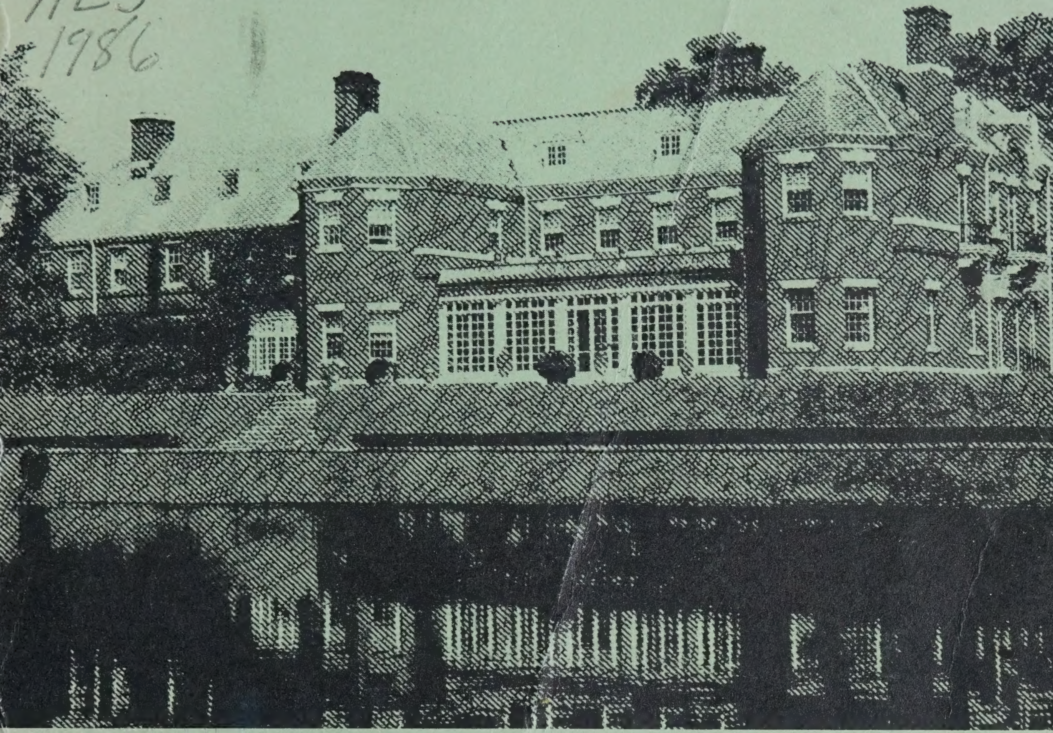
621.3811
AL5
1986

ENGINEERING LIBRARY



Proceedings

TWENTY-FOURTH ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

October 1-3, 1986

Allerton House, Monticello, Illinois
Sponsored by the
Coordinated Science Laboratory and the
Department of Electrical and Computer Engineering of the
University of Illinois at Urbana-Champaign

The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

ENGINEERING

Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.

To renew call Telephone Center, 333-8400

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

PHOTO REPRODUCTION
AUG 27 REC'D
PHOTO REPRODUCTION
SEP 03 REC'D
INTERLIBRARY LOAN
MAR 29 REC'D
INTERLIBRARY LOAN
REC'D OCT 06 '88
INTERLIBRARY LOAN
DUE 11-7-88
REC'D NOV 0
DEC 5 1988
DEC REC'D

MAR 07 1989
REC'D FEB 23 '89
PHOTO REPRODUCTION
NOV 28 REC'D
JUL 6 1990
FEB 05 REC'D
PHOTO REPRODUCTION
AUG 23 REC'D
MAR 05 REC'D
INTERLIBRARY LOAN
MAY 18 2010
SEP 15 1992
AUG 25 REC'D
JUL 21 1993

L161—O-1096

PROCEEDINGS

TWENTY-FOURTH ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

Michael C. Loui
Juraj V. Medanic
Conference Co-Chairman

Conference held
October 1-3, 1986
Allerton House
Monticello, Illinois

sponsored by
The Coordinated Science Laboratory
and
The Department of Electrical and Computer Engineering
of the
UNIVERSITY OF ILLINOIS
at
Urbana-Champaign

Digitized by the Internet Archive
in 2026 with funding from
University of Illinois Urbana-Champaign

https://archive.org/details/proceedings1986alle_0

FOREWORD

The Twenty-Fourth Annual Allerton Conference on Communication, Control, and Computing was held at Allerton House, Monticello, Illinois on October 1, 2, and 3, 1986. The conference was sponsored by the Coordinated Science Laboratory and the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. These *Proceedings* contain the papers presented at the Conference.

We are grateful to Professor James L. Massey of the Swiss Federal Institute of Technology, ETH-Zürich, for his splendid plenary lecture "Cryptography and System Theory." We are also grateful to Professor A. C. Bovik of the University of Texas at Austin for organizing the invited session "Median and Related Filters." We would like to thank all the session chairs and the authors for their contributions.

We appreciate the support of Professor M. E. Van Valkenburg, Dean of the College of Engineering and founder of the series of Allerton Conferences, Professor T. N. Trick, Head of the Department of Electrical and Computer Engineering, and Professor W. K. Jenkins, Acting Director of the Coordinated Science Laboratory. We would also like to thank our colleagues Professors J. A. Abraham, E. Arikan, K. S. Arun, P. Banerjee, M. T. Basar, J. Grizzle, W. K. Fuchs, B. E. Hajek, T. S. Huang, R. Iyer, P. V. Kokotovic, P. R. Kumar, H. Lee, D. C. Munson, J. H. Patel, W. R. Perkins, K. R. Poolla, H. V. Poor, F. P. Preparata, M. B. Pursley, V. Ramachandran, D. V. Sarwate, H. Sira-Ramirez, M. W. Spong, and B. W. Wah for their generous assistance in reviewing papers and organizing the papers into sessions.

We are grateful to Mr. Steven Williams for his handling of the business affairs of the Conference, to Mr. Robert Ebeling and his assistants for their timely printing of large numbers of programs and registration packets, and to Ms. Irene Brown, Ms. Rose Harris, Ms. Tina Lipscomb, Ms. Dixie Murphy, Ms. Rosemary Wegeng, Ms. Carol Willms, and Ms. Dee Wrather for their capable assistance in handling the Conference announcement, registration, program preparation, and editing and sales of these *Proceedings*.

Michael C. Loui

Juraj V. Medanic

TABLE OF CONTENTS

FOREWORD i

PLENARY LECTURE

CRYPTOGRAPHY AND SYSTEM THEORY 1
J.L. Massey

I-A: COMBINATORIAL ALGORITHMS

Chairman: D.J. Brown
University of Illinois

APPROXIMATION ALGORITHMS FOR THE SHORTEST COMMON SUPERSTRING PROBLEM 9
J. Turner

AN EXPERIMENTAL STUDY OF MEDIAN-SELECTION IN QUICKSORT 19
C. McGeoch

TRANSITIVE CLOSURE FOR DYNAMIC GRAPHS 29
G.F. Italiano

HOW HARD IS IT TO PARTITION A PATH? 39
M. Lucertini, Y. Perl, and B. Simeone

A COMPARISON OF LOW COMPLEXITY ALGORITHM FOR FINDING MINIMUM CYCLE CUTSETS 49
H. Levy

AN EFFICIENT ALGORITHM FOR FINDING A MAXIMUM CLIQUE OF AN OVERLAP GRAPH 59
S. Masuda, T. Kashiwabara, K. Nakajima and T. Fujisawa

MAXIMUM ECONOMICAL DOMINATION IN A TREELIKE NETWORK 69
J. McHugh and Y. Perl

I-B: ESTIMATION

Chairman: B. Aazhang
Rice University

STABILITY IN LINEAR ESTIMATION 79
P.A. Kelly and W.L. Root

L*l*-FILTERS 88
F. Palmieri and C.G. Boncelet, Jr.

A PARAMETRIC APPROACH FOR IMPULSE RESPONSE ESTIMATION 90
U.K. Bhargava, R.L. Kashyap and D.M. Goodman

PARAMETER ESTIMATION FOR THE GENERALIZED GAUSSIAN NOISE MODEL 92
M.K. Varanasi and B. Aazhang

MVSE ESTIMATION WITH AN MSE CONSTRAINT 102
J.D. Gibson

ITERATIVE MAXIMUM-LIKELIHOOD ESTIMATION OF TOEPLITZ CONSTRAINED COVARIANCES 111
M.I. Miller, D.L. Snyder and M. Turmon

ADAPTIVE FILTERING FOR A LOCALIZED DISTORTION MEASURE 113
J.D. Gibson and S.D. Gray

LATTICE IMPLEMENTATION OF A NOVEL RECURSIVE ESTIMATION ALGORITHM 115
Y.-F. Huang and A.K. Rao

A NEW ESTIMATION ALGORITHM FOR SYSTEMS WITH DYNAMICAL MULTIPLICATIVE NOISE 125
B.S. Chow and W.P. Birkemeier

I-C: LINEAR SYSTEMS 1

Chairman: K.R. Poolla
University of Illinois

FREQUENCY DOMAIN CONDITIONS FOR DISTURBANCE DECOUPLING BY OUTPUT FEEDBACK 135
R.P. Singh and K.S. Narendra

ZERO SPACES, POLE SPACES, DUALS, and INVERSE SYSTEMS 145
B.F. Wyman and M.K. Sain

SQUARING DOWN BY STATIC AND DYNAMIC COMPENSATORS 150
A. Saberi and P. Sannuti

ASYMPTOTIC FORMS OF THE GENERALIZED HESSENBERG REPRESENTATION 160
D.K. Lindner and D.W. Luse

UNIQUENESS OF THE MINIMAL DYNAMIC COVER AND THE ASSOCIATED SOLUTION TO SYLVESTER'S EQUATION 162
A.F. Assal and I.B. Rhodes

EQUIVALENT CHARACTERIZATIONS OF DETECTABILITY AND STABILIZABILITY OF LINEAR TIME-VARYING SYSTEMS 164
H.-M. Tai

DECENTRALIZED FIXED MODES FOR GENERAL IMPROPER SYSTEMS 194
T.N. Chang and E.J. Davison

THE COMPLETE SET OF STABILIZING CONTROLLERS FOR 2-D MULTIVARIABLE SYSTEMS 204
V. Raman and R. Liu

I-D: DIGITAL SIGNAL PROCESSING

Chairman: K.S. Arun
University of Illinois

MATRIX PRE-CONDITIONING ON OPTICAL LINEAR ALGEBRA PROCESSING 184
A.K. Ghosh

RECONSTRUCTION OF FREQUENCY-OFFSET FOURIER BY ALTERNATING PROJECTION ONTO CONSTRAINT SETS 194
C.A. Weber and J.P. Allebach

PHASE RETRIEVAL AND THE DIGITAL RECONSTRUCTION OF HOLOGRAMS: COMPARISON OF IN-LINE AND SPLIT-BEAM HOLOGRAPHY 204
G. Liu and P.D. Scott

THE USE OF GRENANDER'S SIEVES IN QUANTUM-LIMITED IMAGING 213
D.L. Snyder and M.I. Miller

NEW APPROACHES TO REAL-TIME TRANSFORM CODING AND RECONSTRUCTION OF SIGNALS AND IMAGES 214
O.K. Ersoy and C.H. Chen

SPECTRAL PROPERTIES OF A FINITE FIELD DFT 224
M.W. Marcellin and T.R. Fischer

II-A: SIGNAL DETECTION

Chairman: H.V. Poor
University of Illinois

ROBUST DETECTORS FOR VERY HEAVY TAILED NOISE 226
D. Warren and J.B. Thomas

ASPECTS OF ROBUST SIGNAL DETECTION FROM A DIFFERENTIAL GEOMETRIC VIEWPOINT 236
M.W. Thompson and D.R. Halverson

MINIMAX ROBUST DISCRIMINATION FROM DIFFUSION PROCESSES WITH UNCERTAIN PARAMETERS 246
E. Geraniotis and A. Chau

DETECTION OF WEAK SIGNALS IN A GENERALIZED OBSERVATION MODEL 256
I. Song and S.A. Kassam

THE ANALYSIS OF SOME UNDERSEA NOISE WITH APPLICATION TO DETECTION 266
P.K. Willett and J.B. Thomas

ORDER STATISTIC DETECTOR FOR SIGNAL DETECTION IN GENERALIZED GAUSSIAN NOISE 276
R. Viswanathan and B. Kumar

TIME-SEGMENTED DETECTION OF A RANDOMLY ARRIVING SEMI-INFINITE SIGNAL 284
L.A. Longley and C.C. Lee

A COMPARISON OF ERROR PROBABILITY COMPUTATIONS USING LARGE DEVIATIONS METHODS FOR DETECTORS WITH DEPENDENT DATA 286
K.B. Letaief and J.S. Sadowsky

II-B: DATA STRUCTORS AND GEOMETRIC ALGORITHMS

Chairman: H. Edelsbrunner
University of Illinois

STABLE DUPLICATE-KEY EXTRACTION WITH OPTIMAL TIME AND SPACE BOUNDS 288
B.-C. Huang and M.A. Langston

ARRAY ACCESS BOUNDS FOR BLOCK STORAGE MEMORY SYSTEMS 296
A. Park, K. Balasubramanian and R.J. Lipton

A NOTE ON LOWER BOUNDS FOR MIN-MAX HEAPS 306
A. Hasham and J.-R. Sack

GENERATION OF OPTIMAL BINARY SPLIT TREES 308
J.H. Hester and D.S. Hirschberg

HOW MUCH ON THE AVERAGE IS THE PATRICIA TRIE BETTER? 314
W. Szpankowski

RECOGNIZING POLYGONS: OR HOW TO EAVESDROP 324
J.A. Dean, A. Lingas and J.-R. Sack

IMPROVED BOUNDS FOR RECTANGULAR AND GUILLOTINE PARTITIONS 334
T. Gonzalez and S.-Q. Zheng

II-C: MEDIAN AND RELATED FILTERS

Chairman: A.C. Bovik
University of Texas at Austin

PROPERTIES AND APPLICATIONS OF MEDIAN AND RANK-ORDER FILTERS: AN INTUITIVE DISCUSSION 344
N.C. Gallagher, Jr. and C.A. Weber

THEORETICAL ANALYSIS OF MORPHOLOGICAL FILTERS 353
R.L. Stevenson and G.R. Arce

A CLASS OF RANK FILTERS BASED ON THE WILCOXON SIGNED RANK STATISTIC 363
I. Song and S.A. Kassam

CHOOSING THE STACK FILTER WHICH MINIMIZES THE MEAN ABSOLUTE ERROR CRITERION 373
E.J. Coyle

THE EFFICIENT DESIGN OF ORDER STATISTIC FILTERS 383
C.G. Boncelet, Jr.

THE EQUIVALENCE OF ORDER STATISTIC AND NONRECURSIVE LINEAR DIGITAL FILTERS UNDER MONOTONICITY CRITERIA 390
H.G. Longbotham and A.C. Bovik

SUCCESSIVE PARTITIONING AND SELECTION FILTERS 400
Y.H. Lee, Y. Lee and A.T. Fam

II-D: LINEAR SYSTEMS II

Chairman: Z. Uskokovic
University of Illinois

GENERALIZED STRUCTURAL DECENTRALIZED FIXED MODE DETERMINATION 410
E.J. Davison and W. Gesing

DECENTRALIZED LQG/LTR CONTROLLER DESIGN WITH LOCAL MODELS 420
A. Iftar and U. Ozguner

DISCRETE POLYNOMIAL CONTROL THEORY 428
K. Ossia and H.W. Smith

ON THE EUCLIDEAN CONTROLLABILITY OF INFINITE DELAY SYSTEMS 438
A.S.C. Sinha

FAULT TOLERANT CONTROLLER DESIGN USING MULTIPLE FEEDBACK LOOPS 444
J. Glower and U. Ozguner

II-E: NONLINEAR SYSTEMS AND ALGORITHMS

Chairman: J. Grizzle
University of Illinois

STATE OBSERVATION OF NONLINEAR SYSTEMS: A COMPARATIVE STUDY 446
B.L. Walcott and S.H. Zak

A VARIABLE STRUCTURE CONTROL APPROACH TO THE NONLINEAR DECENTRALIZED PROBLEMS 456
G.P. Matthews and R.A. DeCarlo

DETERMINING STABILITY OF A SLOWLY VARYING NONLINEAR SYSTEM THROUGH A FAMILY OF LINEARIZATIONS 465
C.L. DeMarco and B.R. Barmish

A PARAMETRIZATION OF THE FIRST-ORDER OPTIMALITY CONDITIONS FOR A CLASS OF DISCRETE OPTIMAL CONTROL PROBLEM ... 473
G.G.L. Meyer and L.J. Podrazik

ON LINE SEARCH TERMINATION CRITERIA FOR AND CONVERGENCE OF CONIC ALGORITHMS FOR MINIMIZATION 483
K.A. Ariyawansa

II-F: STOCHASTIC SYSTEMS

Chairman: N.S. Namachchivaya
University of Illinois

CONTROLLED DISCRETE EVENTS GENERATED BY DIFFUSION-THRESHOLD PROCESSES 485
C.J. Conrad, N.H. McClamroch and C.-P. Lee

STABILIZING CONTROL DESIGN FOR A CLASS OF STOCHASTIC NONLINEAR SYTEMS 493
E. Yaz

COMPENSATOR DESIGN FOR STOCHASTIC-PARAMETER SYSTEMS WITH INCOMPLETE STATE INFORMATION 499
E. Yaz

III-A: COMMUNICATION SYSTEMS

Chairman: J.S. Lehnert
Purdue University

THE EFFECT OF LASER LINEWIDTH AND PHASE DISTORTION IN A CASCADED ILL REPEATER SYSTEM ON SYSTEM PREFORMANCE........ 501
V. O'Byrne

EXCEEDANCE DISTRIBUTION FOR THE OUTPUT OF A DELAY AND MULTIPLY RECEIVER WITH GAUSSIAN NOISE INPUT 511
D.E. Reed and M.A. Wickert

RATIO-STATISTIC COMBINING FOR CHANNELS WITH PARTIAL-BAND INTERFERENCE 519
C. Keller

PROBABILITY OF ERROR FOR DSSS MULTIPLE-ACCESS SYSTEMS........ 528
N. Nazari and R.E. Ziemer

MULTIPLE-ACCESS CAPABILITY OF REED-SOLOMON CODED FREQUENCY HOP SPREAD-SPECTRUM COMMUNICATION SYSTEMS 530
B.G. Kim and W. Stark

COLLISON RESOLUTION ALGORITHMS FOR NETWORKS WITH SPREAD-SPECTRUM CAPTURE CAPABILITY 540
M. Soroushnejad and E. Geraniotis

III-B: VLSI LAYOUT AND COMPUTATION

Chairman: F.P. Preparata
University of Illinois

AN APPROXIMATION ALGORITHM FOR ROUTING TWO-TERMINAL NETS AROUND TWO RECTANGLES 550
T.F. Gonzalez and S.-L. Lee

THREE-LAYER CHANNEL ROUTING WITH STACKED PINS 560
M.L. Brady and D.J. Brown

THE ENERGY COMPLEXITY OF TRANSITIVE FUNCTIONS 562
L. Snyder and A. Tyagi

OPTIMAL 3-DIMENSIONAL LAYOUTS OF COMPLETE BINARY TREES 573
A.K. Gupta and S.E. Hambrusch

AN EFFICIENT VLSI ARCHITECTURE WITH APPLICATION TO GEOMETRIC PROBLEMS 575
V.K.P. Kumar and H.M. Alnuweiri

VLSI SUPERCOMPUTING 585
S. Manohar and G. Baudet

CYLINDRICAL ARRAYS FOR MATRIX MULTIPLICATION 595
W.A. Porter and J.L. Aravena

SORTING IN CONSTANT NUMBER OF ROW AND COLUMN PHASES ON A MESH 603
J. Marberg and E. Gafni

III-C: IDENTIFICATION AND ADAPTIVE CONTROL

Chairman: T.A.W. Dwyer
University of Illinois

CONDITIONS FOR DESIGNING STRICTLY POSITIVE REAL TRANSFER FUNCTIONS FOR ADAPTIVE OUTPUT ERROR IDENTIFICATION 613
S. Dasgupta and A.S. Bhagwat

REMARKS ON COMPLEXITY OF SYSTEM IDENTIFICATION 623
G.K. Heitman and W.L. Root

ROBUST ADAPTIVE CONTROLLER 631
K. Hamada, M. Suzuki and Y. Ando

DEADBEAT PARAMETER IDENTIFICATION OF DARMA PROCESSES............ 641
H.F. VanLandingham and M.A. Hopkins

ADAPTIVE CONTROL OF A CLASS OF UNCERTAIN DYNAMIC SYSTEMS..... 643
F. Pourboghrat and A. Datta

ADAPTIVE CONTROL OF DISCRETE TIME-INVARIANT PROCESSES HAVING VARIABLE TIME DELAY 645
H.-S. Yang, Y.-C. Kim and K.-H. Kim

PARAMETER SPACE ANALYSIS OF AN ADAPTIVE SYSTEM 655
F.M.A. Salam

III-D: IMAGE PROCESSING

Chairman: E.J. Delp
Purdue University

RESTORATION OF SMEARED IMAGES SUBJECT TO CAMERA MOTION 665
C.-T. Chen

SPATIO-TEMPORAL FILTERING OF MOVING IMAGE 675
S. Hashimoto, H. Oda and Y. Sato

ROBUST IMAGE MODELLING TECHNIQUES WITH AN IMAGE RESTORATION APPLICATION 685
K.-B. Eom and R.L. Kashyap

ON A MULTIPLE REGISTRATION PROBLEM 695
A. Bruckstein and M. Lindenbaum

ENCODING BIOMEDICAL IMAGES WITH QUADTREES 703
D. Fuhrmann and M. Brown

ADAPTIVE SEGMENTATION OF NOISY IMAGES: AN EM ALGORITHM APPROACH 705
H. Derin and S. Lakshmanan

AN APPLICATION OF TENSOR THEORY TO 3-D SHAPE ANALYSIS............... 707
N. Ansari and E.J. Delp

REAL-TIME PATTERN RECOGNITION WITH DISCRETE FOURIER PREPROCESSING TRANSFORMS 709
O.K. Ersoy and D.Y. Kim

III-E: FOUNDATIONS OF COMPUTING

Chairman: D.E. Muller
University of Illinois

FAIR PROOFS OF DISTRIBUTED PROGRAMS THAT ARE SAFE FROM CONSPIRACY 711
R.K. Shyamasundar

ON GLOBAR VS. LOCAL STATE FAIRNESS 721
R.R. Howell, L. Rosier and H.-C. Yen

THE MEMBERSHIP AND EQUIVALENCE PROBLEMS FOR PICTURE LANGUAGES 730
C. Kim and I.H. Sudborough

AVOIDING LATCH FORMATION IN REGULAR EXPRESSION RECOGNIZERS 740
M.J. Foster

ON THE POWER OF PROBABILISTIC ONE-TAPE TURING MACHINES 749
B. Kalyanasundaram and G. Schnitger

IV-A: PARALLEL COMPUTATION

Chairman: V. Ramachandran
University of Illinois

PARALLEL ALGORITHMS FOR SOME FUNCTIONS OF TWO CONVEX POLYGONS 758
M.J. Atallah and M.T. Goodrich

A PARALLEL ALGORITHM FOR TRIANGULATING SIMPLICIAL POINT SETS IN SPACE WITH OPTIMAL SPEED-UP 768
H. ElGindy

EFFICIENT PARALLEL ALGORITHMS FOR SOLVING SOME TREE PROBLEMS 777
X. He

PARALLEL MATROID INTERSECTION 787
A. Fu and J.G. Peters

SOLVING COMBINATORIAL PROBLEMS ON ARRAYS WITH ONE-WAY DATAFLOW 797
C.D. Savage, M. Stallmann and J.E. Perry

A PARTITIONING PROBLEM ARISING IN THE DESIGN OF SMALL UNIVERSAL EMULATORS 807
R. Newman-Wolfe

ASPECTS OF PARALLEL COMPUTATION IN GLOBAL OPTIMIZATION 812
P.M. Pardalos

IV-B: COMMUNICATION NETWORKS

Chairman: E. Arikan
University of Illinois

A DECOMPOSITION METHOD FOR OPTIMAL LINK SCHEDULING 822
R.G. Ogier

A PERIODIC SCHEDULING PROBLEM IN FLOW-CONTROL FOR DATA COMMUNICATION NETWORKS 824
U. Mukherji

THE ALGEBRAIC AND GEOMETRIC STRUCTURES OF MARKOVIAN PETRI NETWORK LATTICES 834
A.A. Lazar and T.G. Robertazzi

MIXING COLLISION RESOLUTION ALGORITHMS EXPLOITING INFORMATION OF SUCCESSFUL MESSAGES 844
I. Kessler and M. Sidi

INSTABILITY CONDITIONS ARISING IN THE ANALYSIS OF SOME MULTIACCESS PROTOCOLS 854
W. Szpankowski and V. Rego

ANALYSIS OF ACCESS PROTOCOLS FOR A CENTRALIZED NETWORK 864
S. Uppal and J..A. Bucklew

MULTIPLE ACCESS ALGORITHMS VIA GROUP TESTING FOR HETEROGENEOUS POPULATION OF USERS 866
D. Kurtz and M. Sidi

SPLITTING-TYPE ALGORITHMS IN A MULTI-STATION PACKET RADIO NETWORK 868
A. Bar-David and M. Sidi

IV-C: ROBUSTNESS IN FEEDBACK

Chairman: H. Sira-Ramirez
University of Illinois

APPLICATION OF INTERVAL ANALYSIS TECHNIQUES TO THE PARAMETER TOLERANCE PROBLEM FOR LINEAR DYNAMICAL SYSTEMS 870
A.N. Michel and E.P. Oppenheimer

THE GENERAL STRUCTURED SINGULAR VALUE PROBLEM WITH TWO BLOCKS OF UNCERTAINTY 879
J.S. Freudenberg

A CAD METHOD AND ASSOCIATED ARCHITECTURES FOR LINEAR CONTROLLERS 889
S.P. Boyd, V. Balakrishnan, C.H. Barratt, N.M. Khraishi
X. Li, D.G. Meyer and S.A. Norman

THE INADEQUACY OF INPUT-OUTPUT INFORMATION IN ROBUST COMPENSATOR DESIGN 889
J.D. Cobb

A FUNCTIONAL ANALYSIS APPROACH TO AUTOMATIC GAIN CONTROL.. 901
D. Cochran

APPLICATION OF THE ROBUST SERVOMECHANISM CONTROLLER TO SYSTEMS WITH PERIODIC TRACKING/DISTURBANCE SIGNALS 903
E.J. Davison and P. Patel

A MULTI-TIME SCALE CANONICAL FORM AND A HIGH-GAIN REGULATOR PROBLEM 912
Y. Ando, M. Suzuki and K. Hamada

ANALYSIS AND DESIGN OF HIGH-GAIN FEEDBACK SAMPLED-DATA SYSTEMS VIA SINGULAR PERTURBATION APPROACH 914
H. Oloomi and M.E. Sawan

IV-D: FAULT TOLERANCE AND TESTING

Chairman: W.K. Fuchs
University of Illinois

AN EXPERIMENTAL STUDY OF TEST PATTERN AND RESPONSE COMPRESSION TECHNIQUES FOR BIST 916
J.T. King, R. Dandapani and K.K. Saluja

STV-TESTABILITY: AN APPROACH TO TESTING BILATERAL SYSTOLIC ARRAYS 925
H. Elhuni and A. Vergis

ON THE DESIGN OF EMBEDDED TSC CHECKERS 935
S. Kundu and S.M. Reddy

A FAULT-TOLERANT TERNARY CCD SYSCTOLIC ARRAY PROCCESSOR 944
J.-Y. Han

CONCURRENT ERROR DETECTION IN FFT NETWORKS 954
J.-H. Kim

AN APPROACH TO AN OPTIMAL STRATEGY OF RECOVERY POINT INSERTION IN DISTRIBUTED FAULT TOLERANT COMPUTING SYSTEMS 964
G. Cantone and B. Ciciani

IV-E: QUANTIZATION AND DISTRIBUTED DETECTION

Chairman: E. Geraniotis
University of Maryland

A DISTRIBUTED DETECTION SCHEME BASED ON SOFT LOCAL DECISIONS 974
J.J. Chao and C.C. Lee

OPTIMAL DECISION FUSION IN MULTIPLE SENSOR SYSTEMS 984
S.C.A. Thomopoulos, R. Viswanathan and D.P. Bougoulias

DESIGN OF OPTIMAL SAMPLING SCHEMES FOR THE GAUSSIAN HYPOTHESIS TESTING PROBLEM 994
R.K. Bahr and J.A. Bucklew

DESIGN ALGORITHM FOR VARIABLE-LENGTH VECTOR QUANTIZERS 1004
J.-P. Adoul and M. Delprat

CONBTOUR GAIN VECTOR QUANTIZATION OF GAUSS-MARKOV SOURCES 1012
K.T. Malone and T.R. Fischer

ON SOFT BIT ALLOCATION 1021
A. Bruckstein

ARRAY CALIBRATION IN A CORRELATED SIGNAL ENVIRONMENT 1023
T. Lei and C.N. Dorny

V-A: INFORMATION THEORY AND CODING

Chairman: D.V. Sarwate
University of Illinois

CAPACITY REGION OF GAUSSIAN CDMA CHANNELS: THE SYMBOL-SYNCHRONOUS CASE 1025
S. Verdu

DATA COMPRESSION OF A SIMPLE COMPOSITE SOURCE 1035
J.D. Gibson and D.G. Jeong

COMPARISON OF TRELLIS CODED MODULATION SCHEMES USING A LOWER BOUND ON THE FREE EUCLIDEAN DISTANCE 1037
M. Rouanne and D.J. Costello, Jr.

RANDOM BURST CHANNELS AND AN ADAPTIVE BURST ERROR CORRECTING VITERBI ALGORITHM 1047
C.B. Schlegel and M.A. Herro

ON SELF-SYNCHRONIZING ENCRYPTION SYSTEMS 1057
K. Nakamura

V-B: CONCURRENT AND DISTRIBUTED COMPUTATION

Chairman: J.G. Peters
Simon Fraser University

THE SIBLING TRIE: A HIGHLY CONCURRENT SEARCH STRUCTURE 1064
J. Parker and U. Manber

ECONOMICAL ATOMIC MULTI-READER REGISTERS 1073
R. Newman-Wolfe

ON FINDING AND UPDATING SHORTEST PATHS DISTRIBUTIVELY 1079
K.V.S. Ramarao and S. Venkatesan

OPTIMAL SNAPSHOTS AND THE MAXIMUM FLOW IN PRECEDENCE GRAPHS 1089
E. Gafni and V.C. Barbosa

V-C: APPLICATIONS OF CONTROL

Chairman: M.W. Spong
University of Illinois

COMPOSITE CONTROL OF SINGULARLY PERTURBED SYSTEMS WITH APPLICATIONS TO FLIGHT MECHANICS PROBLEMS 1098
Y.-N. Hu and H. Khalil

MODELING OF AN ASSEMBLY TASK WITH TWO COOPERATING MANIPULATORS 1108
M.A. Unseren and A.J. Koivo

DESIGN OF PI-REGULATORS FOR MANIPULATORS VIA SINGULAR PERTURBATION 1117
F. Khorrami and U. Ozguner

A REAL-TIME KNOWLEDGE-BASED CELL CONTROLLER 1126
I. Mina

COMPUTER-AIDED DESIGN OF MULTIVARIABLE CONTROL SYSTEMS WITH APPLICATIONS TO AIRCRAFT FLYING QUALITIES 1128
L.R. Pujara

VELOCITY FEEDBACK CONTROL AND THE DYNAMIC PROPERTIES OF ROBOT MANIPULATORS 1130
F.M.A. Salam

V-D: COMPUTER ARCHITECTURE

Chairman: P. Banerjee
University of Illinois

MEMORY REDUCTIONS FOR ROM BASED LOGARITHMIC CONVERSION...... 1132
S.C. Lee and T. Migimatsu

A MINIMAL SPANNING TREE BASED ALGORITHM FOR DELAY-BOUNDED ARRAY RECONFIGURATION 1134
D. Sciuto

PRELIMINARY RESULTS ON MULTIPROCESSOR MODELING AND ANALYSIS USING STOCHASTIC HIGH LEVEL PETRI NETS 1136
D.C. Marinescu and C. Lin

V-E: CONTROL AND MODELING OF MARKOV PROCESSES

Chairman: T.U. Basar
University of Illinois

SAMPLE-PATH CONSTRAINT FOR MARKOV DECISION PROCESSES - A NEW CRITERION 1138
K.W. Ross and R. Varadarajan

OPTIMAL PRIORITY ASSIGNMENT WITH GENERAL CONTRAINTS 1147
E. Altman and A. Shwartz

DYNAMIC SCHEDULING OF THREE QUEUES IN TANDEM 1149
R.A. Milito and D. Cansever

GEOMETRIC AGGREGATION, EXPANSION AND PARTIAL OBSERVATIONS FOR MARKOV PROCESSES 1159
J.M. Ferrandiz and A.A. Lazar

A NONROBUSTNESS RESULT ON THE RATE OF CONVERGENCE FOR MARTINGALES 1169
J.M. Morrison and G.L. Wise

CRYPTOGRAPHY AND SYSTEM THEORY

JAMES L. MASSEY
Institute for Signal and Information Processing
Swiss Federal Institute of Technology
CH-8092 Zurich, Switzerland

ABSTRACT

A series of instances is given to show that system theory is a useful tool in cryptography, particularly in the study of stream ciphers. The one-time pad, which provides perfect secrecy, is shown to be a time-invariant linear system with an infinite-dimensional state space. Perfect linear ciphers with a finite dimensional state space are then constructed and a conjecture on the minimum amount of secret key needed is given. The cryptographic concept of linear complexity is shown to be completely system-theoretic and closely related to the discrete Fourier transform; its cryptographic significance is discussed. Siegenthaler's concept of correlation-immune combining functions for running key generators is given a system-theoretic formulation in terms of Walsh transforms.

INTRODUCTION

In Fig. 1, we show the model of a "general secrecy system" proposed by Shannon [1] in 1949. [With no loss of essential generality, we shall hereafter assume that the plaintext digits x_i, the ciphertext digits y_i, and the components of the secret key K are all binary digits, i.e., elements of the finite field GF(2).] The purpose of the encrypter is to transform the plaintext, under control of the secret key, into ciphertext in such a way that it is easy to recover the plaintext from the ciphertext if one knows the secret key, but that it is impossible, or unfeasibly difficult, to recover the plaintext from the ciphertext if one does not know the secret key.

Surely the greatest surprise in the recent history of cryptography was the demonstration by Diffie and Hellman [2] in 1976 that secrecy systems could be built that dispensed with the "secure channel" for transmitting a secret key to the destination. In Shannon's work [1], we find no premonition of such "public-key secrecy systems" and we must today say that his Fig. 1 gives the model of only a "general *secret-key* secrecy system".

Public-key cryptography has certainly been the glamor field of cryptographic research in the past decade, but secret-key systems continue to dominate practice. Indeed, as understanding of public-key methods has crystallized, it has become clear that the main use of public-key techniques (at least in the forseeable future and possibly forever) will be for the exchange of keys that will then be used in secret-key systems for securing the actual message traffic. In a sense, public-key methods will be used to implement the "secure channel" of Fig. 1. The main reason for this prognosis is that the secure public-key schemes thus far proposed are computationally so intensive that data rates of at most a few kilobits per second can be accommodated, dismally far below the demanded data rates in most applications.

This has been a rather long apologia for our decision to treat only secret-key systems in this paper. In our opinion, secret-key systems will always be the workhorses of cryptography. Moreover, secret-key systems are much more amenable to a system-theoretic treatment than are public-key systems, which rest on a foundation of computational complexity.

Linearity, which is often described as "the curse of the cryptographer", is perhaps the greatest blessing of the system theorist. The cryptographer usually tries to make the enciphering process as nonlinear as possible. It might seem then that "classical" linear system theory would have little to say about cryptography. Our aim in this paper is to demonstrate the contrary. We give a succession of examples where the techniques of linear system theory have proved exceedingly useful in cryptography. For these examples, we have drawn heavily from our own experience, but their lessons are general.

THE ONE-TIME PAD

Shannon [1] has defined a secrecy system to be *perfect* if the ciphertest is statistically independent of the plaintext, i.e., if the enemy cryptanalyst in Fig. 1 can do no better than to guess at the plaintext without bothering to observe the ciphertext. And Shannon seems to have been the first to point out explicitly that the "one-time pad" (or modulo-two "Vernam system") of Fig. 2 is a perfect cipher. The binary symmetric source (BSS), which serves as the key source in Fig. 2, is an ideal random number generator that emits a binary "coin-tossing sequence" $z_0, z_1, z_2, \cdots$. For any n and any choice of the plaintext, the first n digits of the ciphertext $(y_0, y_1, \ldots, y_{n-1}) = (x_0, x_1, \ldots, x_{n-1}) + (z_0, z_1, \ldots, z_{n-1})$ are equally likely to form any of the 2^n binary sequences of length n, which establishes the statistical independence between the plaintext and the ciphertext. The one-time pad is indisputably an unbreakable cipher. What keeps the one-time pad from being the universal solution to the secrecy problem is, of course, the vast amount of secret key that it requires to be generated and sent over the secure channel, one bit of secret key for each bit of ciphertext. In fact, as Shannon also showed, the one-time pad uses the very least secret key of any cipher that provides perfect secrecy independently of the plaintext statistics. The cryptographer who must operate with a limited amount of secret key must abandon the hope of achieving perfect secrecy.

It is interesting to consider the one-time pad from a system-theoretic viewpoint. The encrypting system (which coincides with the decrypting system) is probably best understood as a time-invariant linear system [over the field GF(2)] with output equation

$$y_t = x_t + z_t \qquad t = 0,1,2$$

and state

$$s_t = (z_t, z_{t+1}, z_{t+2}, \ldots).$$

The zero-state response is described by the trivial identity transformation

$$y_t = x_t,$$

and the next-state function is an equally trivial left shift of the semi-infinite state vector. The zero-input response is just this state vector itself. As seen by a system theorist, the one-time pad is a linear system in which the response to the input (i.e., to the plaintext) is hidden by the state response, the completely random choice of the initial state (i.e., the secret key) providing perfect secrecy. Linearity may be the curse of the cryptographer, but it is also his best guide to perfect secrecy.

In fact, many (if not most) practical ciphers (i.e., ciphers with a finite key) are designed to mimic the one-time pad so far as possible. These are the so-called additive stream ciphers, the model for which is shown in Fig. 3. The key K is now a binary sequence of length k, which is used to control the running key generator (RKG). The output sequence $z_0, z_1, z_2, \ldots$ of the RKG is called the running key and is added to the plaintext in the same manner as is the secret key in the one-time pad. The output (ciphertext) of the encrypting system may still be decomposed as the sum of the responses due to the input (plaintext) and due to the state, the input response is still governed by the identity transformation $y_t = x_t$, but the state response is always chosen to be a highly nonlinear function of the state. Yet, as we shall soon see, linear system theory has much to say about such additive stream ciphers.

LINEAR CIPHERS

Although the one-time pad can be described as a linear system as we have just shown, most cryptographers would object to calling the enciphering process "linear" as the necessary state space is infinite-dimensional. It is more in keeping with cryptographic tradition to reserve the term *linear cipher* for an enciphering system that is linear and has a finite-dimensional state space, say of dimension k. Are there perfect linear ciphers? The answer is both yes and no. The answer is no if one demands that the enciphering be perfect independently of the input statistics -- the all-zero plaintext must produce one of the 2^k output sequences that are the zero-input responses to the 2^k possible initial states. This the plaintext and ciphertext cannot be statistically independent if the all-zero sequence is a possible plaintext sequence.

On the other hand, suppose that the linear encrypter has finite input memory M so that ciphertext is described by

$$y_t = x_t + h_1(t)x_{t-1} + \cdots + h_M(t)x_{t-M}, \quad t \geqslant 0 \tag{1}$$

where $x_{-1},\ldots,x_{-M}$ form the initial state. We have chosen the coefficient of the current input digit x_t in this time-varying linear recursion to be 1 to meet the requirement that the enciphering be invertible. Suppose further that there is a *plaintext restriction* that ensures that no M consecutive plaintext digits can all be zeroes (as could be achieved by "bit stuffing" with little added redundancy when M is large), where we count $x_{-1},\ldots,x_{-M}$ as plaintext digits for this purpose. If each binary digit $h_i(t)$, $t \geqslant 0$, $1 \leqslant i \leqslant M$, is chosen by an independent coin toss, then it is easy to see from (1) that y_t is equally likely to be 0 or 1, independently of the plaintext digits and of the previous ciphertext digits. Thus plaintext and ciphertext are independent and the secrecy system is perfect! But now we need M bits of secret key [to choose the M coefficients $h_1(t), h_2(t),\ldots,h_M(t)$] for each bit of plaintext. Is this the least possible? The answer is no. We showed elsewhere [3] (together with R.A. Rueppel) that only two bits of secret key per bit of plaintext are needed for perfect secrecy if one chooses the recursion coefficients to satisfy

$$h_{i+2}(t+1) = h_i(t) \qquad i = 1,2,\ldots,M-2 \tag{2}$$

so that only $h_1(t)$ and $h_2(t)$ are "new" random digits at time t. We also conjectured that, when $M > 1$ (as is required to make the plaintext restriction nontrivial), then this is the least amount of secret key in any perfect linear cipher [3]. This conjecture is still open. A counterexample might not only suggest interesting new ciphers, but would also identify a new smallest ensemble of convolutional codes for which Shannon's random coding bound could be proved. In fact, out "two bits of key" solution was borrowed from an earlier construction we had made for such "small" code ensembles [4].

There is an interesting way to look upon the recursion (1) when the coefficients satisfy (2). One can think of these coefficients as being read from the key source (the BSS) into a shift-register of length M that shifts at twice the speed of another shift-register into which the plaintext digits are read, the terms added to x_t in (1) then being the "dot product" of the contents of these shift-registers at time t. The effect of the coefficients is to randomize the contribution from the plaintext digits. It is then a simple step to deduce, as we did in [3], that a good RKG for an additive stream cipher might be built as in Fig. 4 where a simple linear feedback shift-register (LFSR) is used to generate a "pseudo-random" sequence that replaces the BSS output and a second LFSR is used to enforce the plaintext restriction. The former LFSR is shifted twich (or more times) as fast as the latter, and the initial contents of both LFSR's are determined by the secret key. This seems to be a very good RKG (at least no one has as yet reported to us any weaknesses) as well as an excellent random number generator. But our aim in describing this RKG was only to show how a practical secrecy system grew out of entirely linear-system-theoretic considerations.

LINEAR COMPLEXITY

One of the most useful concepts in the study of stream ciphers is that of linear complexity. The *linear complexity* of the sequence $z^n = (z_0, z_1, \ldots, z_{n-1})$ is defined as the smallest nonnegative integer L such that there exists constants $c_1, c_2, \ldots, c_L$ for which

$$z_t + c_1 z_{t-1} + \ldots + c_L z_{t-L} = 0, \quad L \leqslant t < n. \tag{3}$$

[This definition holds for sequences over any field.] Equivalently, L is the length of the shortest LFSR, as shown in Fig. 5, that can generate the sequence z^n when its first L digits form the initial contects of the register.

To see why linear complexity is interesting for stream cirphers, we need to consider a so-called "known plaintext attack" by the enemy cryptanalyst. Frequently, this cryptanalyst knows, or can reliably guess, a considerable portion of the plaintext. For an additive stream cipher, we see from Fig. 3 that this knowledge is equivalent to knowing the corresponding portion of the running key sequence, as the enemy cryptanalyst always knows the ciphertext. For an additive stream cipher to be secure, it is thus necessary that knowledge of a considerable portion of the running key, say of z^n , should not permit the cryptanalyst to predict reliably future bits in the running key. But, if the semi-infinite sequence $(z_0, z_1, z_2, \ldots)$ has linear complexity L, then there is an efficient algorithm, the LFSR synthesis algorithm [5] or so-called "Berlekamp-Massey algorithm" that, when applied to z^n for any $n \geqslant 2L$, finds the shortest generating LFSR after about only nL bit operations and thus enables the cryptanalyst to predict perfectly the future bits of the running key. It follows then that a necessary (but by no means sufficient) condition for a RKG to be secure is that it always produce an output sequence with very large linear complexity.

From a system theory viewpoint, the linear complexity of 2^n is just the minimum dimension of the state space of the autonomous (i.e., no input) linear systems that can produce z^n as the first n digits of their output sequence for some choice of the initial state. The LFSR of Fig. 5 is just a cononical form for the realization of such autonomous linear systems.

There is a curious feature of linear systems over a finite field that seems to have no counterpart for linear systems over an infinite field, namely that the output sequence of any finite-state autonomous system can be generated also by some time-in-variant autonomous linear system over the finite field with a finite-dimensional state space. The reason is that, if the finite state system has N states, then after a transient of duration T digits, where $0 \leqslant T < N$, the output sequence must become periodic with period P where $P \leqslant N-T$. But such an ultimately periodic output sequence can be produced by the LFSR of Fig. 5 with $L = T + P \leqslant N$ in which $c_p = -1$ is the only non-zero feedback coefficient. This explains why linear complexity has a kind of universal significance in cryptography -- every sequence that can be produced by a finite-state machine has finite linear complexity.

Linear complexity has a close connection to another friend of the system theorist, the discrete Fourier transform (DFT). If α is a primitive n-th root of unity in any field F, then the DFT of the F-ary sequence $\underline{b} = (b_0, b_1, \ldots, b_{n-1})$ is the F-ary sequence $\underline{B} = (B_0, B_1, \ldots, B_{n-1})$ where

$$B_i = \sum_{j=0}^{n-1} b_j \alpha^{-ij}. \tag{4}$$

The inverse DFT relation is

$$b_j = \frac{1}{n} \sum_{i=0}^{n-1} B_i \alpha^{+ij}. \tag{5}$$

If we now use (4) to define B_i for all $i \geqslant 0$ (which implies that $B_{i+n} = B_i$), then the Hamming weight (i.e., the number of non-zero components) of $\underline{b}$ is the linear complexity of the semi-infinite sequence $B_0, B_1, B_2, \ldots$. This follows immediately from (4), which displays this sequence as the solution of a constant coefficient linear recursion in which α^{-1} is a root of the characteristic polynomial of the recursion if and only if $b_j \neq 0$. This relation between linear complexity and the DFT has had a profound impact on coding theory, cf. the spectral theory of cyclic codes given by Blahut [6], and is just beginning to find uses in cryptography. As the relationship holds for *all* fields, one suspects that other interesting applications are awaiting it elsewhere in system theory.

CORRELATION-IMMUNITY

A favorite trick of the RKG designer is to combine the output sequence of several LFSR's, each of which has linear complexity at most equal to the length of the LFSR producing it, by means of a highly nonlinear function to produce a running key of large linear complexity. For instance,

$$f(x_1, x_2, x_3, x_4) = x_1 + x_2x_3 + x_3x_4 + x_4 \tag{6}$$

is a combining function for four such sequences. When written in GF(2) sum-of-products form as in (6), the maximum number of factors in a product term is the so-called *nonlinear order* of the combining function, which is 2 for our example. Rueppel and Staffelbach [7] recently proved that, under weak conditions (which are fulfilled in particular when the combined LFSR's have relatively prime lengths, each has an irreducible connection polynomial $C(D) = 1 + c_1D + \cdots + c_LD^L$ of degree equal to its length L, and each has a non-zero initial state), then the linear complexity L of the output sequence can be found by treating the GF(2) sum-of-products expression as an expression over the integers and replacing x_i by the length L_i of the i-th LFSR. For our example, with $L_1 = 29$, $L_2 = 30$, $L_3 = 31$ and $L_4 = 37$, we would obtain

$$L = 29 = (30)(31) + (31)(37) + 37 = 2143.$$

The arguments used by Rueppel and Staffelbach to prove this result are fundamentally system-theoretic.

Siegenthaler noted a few years ago that many combining functions tend to "leak" in the sense that the running key is correlated with one or more of the combined sequences and that such ciphers could be easily broken, even with a "ciphertext only" attack [8]. Siegenthaler went on to suggest one of the most interesting recent notions in stream cipher theory, that of correlation-immunity [9]. The combining function $f(x_1,...,x_n)$ is said to be m-th order correlation-immune if, when $x_1,x_2,...,x_n$ are independent coin-tossing random variables, then the random variable $z = f(x_1,...,x_n)$ is statistically independent of every subset of m or fewer of the random variables $x_1,...,x_n$. Siegenthaler showed further that there is a tradeoff between linear complexity of the output sequence and correlation-immunity of the combining function in the sense that, if the function is correlation immune of order m, then its GF(2) sum-of-products form can have no products with more than n-m factors, i.e., the nonlinear order can be at most n-m

Correlation-immunity also has an intereting system-theoretic formulation. Letting the values of $f(x_1,...,x_n)$ be treated as the real numbers 0 and 1, we can define the *Walsh transform* of the combining function by

$$F(\underline{w}) = \sum_{x_1=0}^{1} \cdots \sum_{x_n=0}^{1} f(x)\,(-1)^{\underline{x}\cdot\underline{w}} \tag{7}$$

where $\underline{w} = (w_1,...,w_n)$ is a vector over GF(2), where $\underline{x} = (x_1,...,x_n)$, and where $\underline{x}\cdot\underline{w} = x_1w_1 + \cdots + x_nw_n$ is evaluated in GF(2) but then treated as a real number in the exponent. Together with G.-Z. Xiao, we showed recently [10] that $f(\underline{w})$ is m-th order correlation-immune if and only if $F(\underline{w}) = 0$ for all $\underline{w}$ with Hamming weight between 1 and m exclusively. The reader can easily check for the example of (6) that $F(1,0,0,1) = F(0,1,0,0) = F(0,0,1,0) = F(0,0,0,1) = 0$ (but $F(1,0,0,1) = -4$) so that this combining function is first-order (but not second-order) correlation-immune. The argument in our proof is based on the easily verified fact that $z = f(x_1, \ldots, x_m)$ is independent of any subset of m or fewer of the independent random variables $x_1,...,x_n$ if and only if z is independent of the single random variable $\underline{x}\cdot\underline{w}$ for every choice of w with Hamming weight between 1 and m.

A FINAL REMARK

We have tried to show enough instances where system theory has been applied to cryptography to convince the reader that crytography should be attractive to system theorists seeking new areas to conquer. The reader can find many further such examples in Rueppel's recent book [11] about stream ciphers. We don't mean to imply that system theory has no applications in the study of block ciphers (such as the Data Encryption Standard), but that is a story for another day.

REFERENCES

[1] C. E. Shannon, "Communication Theory of Secrecy Systems", *B.S.T.J.*, vol. 28, pp. 656-715, Oct. 1949.

[2] W. Diffie and M. E. Hellman, "New Directions in Cryptography", *IEEE Trans. Info. Th.*, vol. IT-22, pp. 644-654, Nov. 1976.

[3] J. L. Massey and R. A. Rueppel, "Linear Ciphers and Random Sequence Generators with Multiple Clocks", pp. 74-87 in *Advances in Cryptology* (Eds. T. Beth, N. Cot and I. Ingemarsson), Proc. Eurocrypt '84. Heidelberg and New York: Springer, 1985.

[4] J. L. Massey, "Error Bounds for Tree Codes, Trellis Codes, and Convolutional Codes with Encoding and Decoding Procedures", pp. 2-57 in *Coding and Complexity* (Ed. G. Longo), CISM Courses and Lectures No. 216. Vienna and New York: Springer, 1976.

[5] J. L. Massey, "Shift-Register Synthesis and BCH Decoding", *IEEE Trans. Info. Th.*, vol. IT-15, pp. 122-127, Jan. 1969.

[6] R. E. Blahut, *Theory and Practice of Error Control Codes*. Reading, Mass.: Addision-Wesley, 1983

[7] R. A. Rueppel and O. J. Staffelbach, "Products of Linear Recurring Sequences with Maximum Complexity", to appear in *IEEE Trans. Info. Th.*

[8] T. Siegenthaler, "Decrypting a Class of Stream Ciphers Using Ciphertext Only", *IEEE Trans. Comp.*, vol. C-34, pp. 81-85, Jan. 1985.

[9] T. Siegenthaler, "Correlation-Immunity of Nonlinear Combining Functions for Cryptographic Applications", *IEEE Trans. Info. Th.*, vol. IT-30, pp. 776-780, Oct. 1984.

[10] G. Z. Xiao and J. L. Massey, "A Spectral Characterization of Correlation-Immune Combing Functions", submitted to *IEEE Trans. Info. Th.*, Aug. 1985.

[11] R. A. Rueppel, *Analysis and Design of Stream Ciphers*. Heidelberg and New York: Springer, 1986

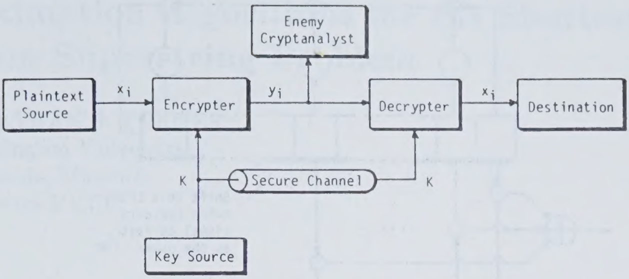

Enemy Cryptanalyst
Plaintext Source
x_i
Encrypter
y_i
Decrypter
x_i
Destination
K
Secure Channel
K
Key Source

Fig. 1: A General (Secret-Key) Secrecy System

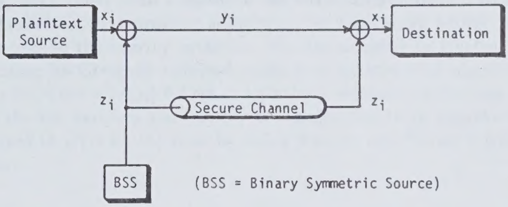

Plaintext Source
x_i
y_i
x_i
Destination
z_i
Secure Channel
z_i
BSS
(BSS = Binary Symmetric Source)

Fig. 2: The One-Time Pad Secrecy System

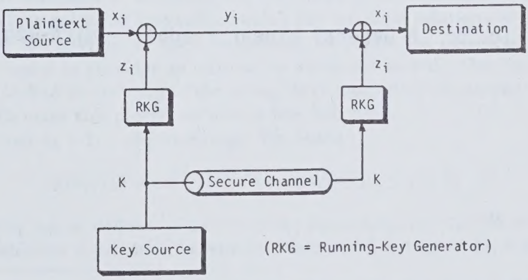

Plaintext Source
x_i
y_i
x_i
Destination
z_i
z_i
RKG
RKG
K
Secure Channel
K
Key Source
(RKG = Running-Key Generator)

Fig. 3: A General Additive Stream Cipher

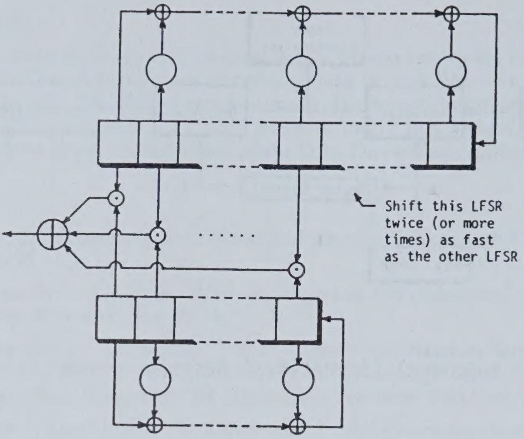

Shift this LFSR
twice (or more
times) as fast
as the other LFSR
......

Fig. 4: A Running-Key Generator Suggested by a Perfect Linear Cipher

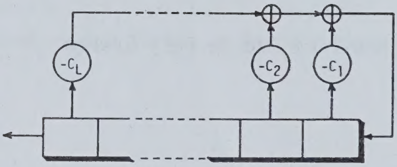


Fig. 5: A General Linear Feedback Shift-Register

Approximation Algorithms for the Shortest Common Superstring Problem

JONATHAN S. TURNER
Washington University
St. Louis, Missouri
jst@wucs.UUCP

Abstract

The object of the shortest common superstring problem (SCS) is to find the shortest possible string that contains every string in a given set as substrings. As the problem is NP-complete, approximation algorithms are of interest. The value of an approximate solution to SCS is normally taken to be its length, and we seek algorithms that make the length as small as possible. A different measure is given by the sum of the overlaps between consecutive strings in a candidate solution. When considering this measure, the object is to find solutions that make it as large as possible. These two measures offer different ways of viewing the problem. While the two viewpoints are equivalent with respect to optimal solutions, they differ with respect to approximate solutions. We describe several approximation algorithms that produce solutions that are always within a factor of two of optimum with respect to the overlap measure. We also describe an efficient implementation of one of these, using McCreight's compact suffix tree construction algorithm. The worst-case running time is $O(m+n \log n)$ for small alphabets, where m is the sum of the lengths of all the strings in the set and n is the number of strings. For large alphabets, the algorithm can be implemented in $O(m \log m)$ time by using Sleator and Tarjan's lexicographic splay tree data structure.

1. Introduction

Let $s_1 = a_1 \ldots a_r$ and $s_2 = b_1 \ldots b_s$ be strings over some finite alphabet Σ. We say that s_1 is a *substring* of s_2 if there is an integer $i \in [0, s-r]$ such that $a_j = b_{i+j}$ for $1 \leq j \leq r$. We also say in this case that s_2 is a *superstring* of s_1.

An instance of the *shortest common superstring problem* (SCS) is a set of strings $S = \{s_1, \ldots, s_n\}$ over a finite alphabet Σ. The object of the problem is to find a minimum length string that is a superstring of every $s_i \in S$. We let $\phi^*(S)$ denote the length of a minimum length superstring.

We say that a set of strings is *substring free* if no string in the set is a substring of any other. We will generally limit our attention to substring free sets. This involves no loss of generality, since any set of strings has a unique substring free subset which has the same solutions as the original set.

We have presented the problem in the conventional way, with the object being to minimize the solution length. It is useful to consider an alternative viewpoint as well. One can view the object of the problem as being to find an ordering of the strings that maximizes the amount of overlap between consecutive strings. To make this precise we need a few definitions.

Let $s_1 = a_1 \ldots a_r$ and $s_2 = b_1 \ldots b_s$ be strings. We define

$$\psi(s_1, s_2) = \max \{k \geq 0 \,|\, a_{r-k+i} = b_i \; 1 \leq i \leq k\}$$

If $\psi(s_1, s_2) = k$ then $s_1 \circ s_2$ is defined to be the string $a_1 \ldots a_r b_{k+1} \ldots b_s$. We note that if s_1, s_2, s_3 are strings, none of which is a substring of another, then $s_1 \circ (s_2 \circ s_3) = (s_1 \circ s_2) \circ s_3$; that is, the

---

The research described here was supported by the National Science Foundation (grant #DCR-8409435) and the National Institutes of Health (grant #RR-01380)

overlapping operation is associative for substring free sets. Consequently, we may write $s_1 \circ s_2 \circ \cdots \circ s_n$ with no ambiguity.

Let π be a permutation on $\{1,\ldots,n\}$. We will usually write π_i for $\pi(i)$. We define

$$\psi_\pi(s_1,\ldots,s_n) = \sum_{i=1}^{n-1} \psi(s_{\pi_i}, s_{\pi_{i+1}})$$

and $\phi_\pi(s_1,\ldots,s_n) = |s_{\pi_1} \circ \cdots \circ s_{\pi_n}|$. Note that for any instance $S = \{s_1,\ldots,s_n\}$ of SCS,

$$\phi_\pi(S) = \|S\| - \psi_\pi(S)$$

where $\|S\| = \sum_{i=1}^n |s_i|$. In particular,

$$\phi^*(S) = \|S\| - \psi^*(S) \quad \text{where} \quad \psi^*(s_1,\ldots,s_n) = \max_\pi \psi_\pi(s_1,\ldots,s_n)$$

Hence, we can view the object of the SCS problem as being to find a mapping π that maximizes ψ_π. Let A be an algorithm for SCS which given a collection of strings $S = \{s_1,\ldots,s_n\}$ produces a mapping $\pi = \pi_A(S)$. We define $\psi_A(S) = \psi_\pi(S)$ and $\phi_A(S) = \phi_\pi(S)$.

SCS was shown to be NP-complete by Maier and Storer in [9]. Another, and more elegant proof appears in [4] and [5]. One obvious application for the problem is data compression. Storer and Szymanski [16] for example, consider a fairly general model of data compression which includes SCS as an important special case. See also [10]. Another application is to DNA sequencing. SCS is one of the simplest models for the problem of recovering DNA sequencing information from experimental data [6,13,15]. To our knowledge the only approximation algorithm to be discussed in the literature is a simple greedy algorithm which is treated briefly by Gallant in [5]. Gallant claims that for this algorithm, which we refer to as SGREEDY, $\phi_{\text{SGREEDY}}(S) \leq (3/2)\phi^*(S)$ for all collections of strings S. We show that this is not in fact true by displaying a set of strings S for which $\phi_{\text{SGREEDY}}(S) \approx 2\phi^*(S)$. We have found no worse example problem than this, but have also been unsuccessful in proving an upper bound on the performance of this algorithm in terms of the length measure. On the other hand, we do show that $\psi^*(S) \leq 2\psi_{\text{SGREEDY}}(S)$.

In section 2 we relate SCS to the longest path problem (LPP) in graphs by describing a transformation from SCS to LPP that preserves solution values with respect to the overlap measure. We then construct three approximation algorithms for LPP, two based on matching and the third a greedy heuristic. By virtue of the transformation from SCS, all three are also approximation algorithms for SCS. We show that the greedy heuristic for LPP always produces solutions within a factor of three of the optimum value. In section 3, we show that the instances of LPP that result from our transformation from SCS have a special structure that allows us to obtain a tighter bound. We also describe an efficient implementation of this greedy algorithm for strings using a compact representation of suffix trees. In section 4, we relate SCS to the traveling salesman problem (TSP) by another transformation that preserves solution values, this time with respect to the length measure. The instances of TSP arising from this transformation are asymmetric, but satisfy the triangle inequality. There are no approximation algorithms known for this problem with provably good worst-case performance, nor have we succeeded in finding any. Nevertheless, this transformation means that if such an algorithm is found, it can be used for SCS as well as TSP. If on the other hand, it turns out that approximating this version of TSP is hard, then any approximation algorithm for SCS, will have to make use of special structural properties present in the instances of TSP that arise from this transformation.

2. SCS and the Longest Path Set Problem

In this section we relate SCS to the *longest path problem* (LPP) in graphs. An instance of the longest path problem is a complete directed graph $G = (V, E)$ with each edge (u, v) having a non-negative integer length $\ell(u, v)$. The *length of a path* p in G is defined to be the sum of the lengths of its edges and is denoted $\lambda_p(G, \ell)$. The object of the longest path problem is to find a Hamiltonian path p (that is a path including every vertex) in G that maximizes $\lambda_p(G, \ell)$. The length of such a longest path is denoted $\lambda^*(G, \ell)$.

Let $S = \{s_1,\ldots,s_n\}$ be an instance of SCS. We define LPP(S) to be an instance (G, ℓ) of LPP with

$$\begin{aligned} V &= \{u_1,\ldots,u_n\} \qquad E = V \times V \\ \ell(u_i, u_j) &= \psi(s_i, s_j) \qquad 1 \leq i,j \leq n,\ i \neq j \end{aligned}$$

```
function edgeset MATCH(digraph $G = (V, E)$, edgelengths $\ell$)
    edgeset $P$, $M$;
    $P := \emptyset$;
    do $E \neq \emptyset \rightarrow$
        $M :=$ MAXMATCH($G, \ell$);
        $P := P \cup M$;
        for $(u, v) \in M \rightarrow$
            Delete from $G$, all edges of the form $(u, x)$ or $(y, v)$;
            Collapse $u$ and $v$ into a single vertex;
        rof;
    od;
    return $P$;
end;
```

Figure 1: Matching Algorithm for LPP

Let π be a permutation on $\{1, \ldots, n\}$. We can view π as defining a Hamiltonian path $u_{\pi_1}, \ldots, u_{\pi_n}$ in G. We let $\lambda_\pi(G, \ell)$ denote the length of this path. We now state a trivial, but useful theorem.

THEOREM 1. *Let $S = \{s_1, \ldots, s_n\}$ be an instance of* SCS, *$(G, \ell) =$* LPP*(S) and let π be any permutation on $\{1, \ldots, n\}$. $\lambda_\pi(G, \ell) = \psi_\pi(S)$. In particular, $\lambda^*(G, \ell) = \psi^*(S)$.*

The theorem implies that any approximation algorithm for LPP is an approximation algorithm for SCS with respect to the overlap measure. In the remainder of this section, we present three simple approximation algorithms for LPP.

Matching Algorithm. A *matching* in a graph $G = (V, E)$ is a set of edges, no two of which share a common vertex. A maximum matching in a graph with edge lengths $\ell(e)$ is a matching M that maximizes $\ell(M)$. We define $\mu^*(G, \ell) = \max_M \ell(M)$ to be the value of a maximum matching. There are algorithms for finding maximum matchings having running times of $O(n^3)$ (where $n = |V|$) [17].

Our first algorithm for LPP is based on the observation that any matching for an instance (G, ℓ) of LPP can be extended to a path (since G is assumed to be complete) and a maximum matching must have total length at least half that of a longest path. (Recall that we are restricting attention to non-negative weights.)

THEOREM 2. *If $(G = (V, E), \ell)$ is an instance of* LPP *then $\lambda^*(G, \ell) \leq 2\mu^*(G, \ell)$.*

Theorem 2 provides the basis for our first approximation algorithm shown in Figure 1. The procedure MATCH starts by finding a maximum matching in G, then removes edges that are ruled out by the selected edges, collapses the selected edges into single vertices and then repeats the process on the new graph. To see that the algorithm does construct a Hamiltonian path, note the following: (1) the edge eliminations ensure that the set P never contains two edges leaving a common vertex or entering a common vertex, (2) the collapsing of edges into single vertices prevents creation of cycles and (3) since the original graph is assumed to be complete, the algorithm will halt only when a complete Hamiltonian path has been constructed.

Theorem 2 implies that $\psi^*(G, \ell) \leq 2\psi_{\text{MATCH}}(G, \ell)$ for any instance (G, ℓ) of LPP. The running time of MATCH is determined primarily by the matching algorithm used. Assuming a matching algorithm that runs in $O(n^3)$ time, we get a running time of $O(n^3 \log n)$ for MATCH.

Directed Matching Algorithm. A *directed matching* in a digraph $G = (V, E)$ is a set of edges, no two of which enter a common vertex and no two of which leave a common vertex. In other words, it is a subgraph of G comprising a collection of disjoint paths and cycles. A maximum directed matching in a graph G with edge lengths $\ell(e)$ is a directed matching M that maximizes $\ell(M)$. We define $\delta^*(G, \ell) = \max_M \ell(M)$ to be the value of a maximum directed matching (where in this case, M ranges over all directed matchings of G). There are algorithms for finding maximum directed matchings with running times of $O(n^{5/2})$ [17].

Given any matching M, let M^- be a subset of M obtained by discarding a least cost edge from each cycle in M. Our next algorithm for LPP is based on the observation contained in the next theorem.

```
function edgeset DIMATCH(digraph $G = (V, E)$, edgelengths $\ell$)
    edgeset $P$, $M$;
    $P := \emptyset$;
    do $E \neq \emptyset \rightarrow$
        $M :=$ MAXDIMATCH$(G, \ell)$;
        $M^- := M -$ one least cost edge from each cycle of $M$;
        $P := P \cup M^-$;
        for each path $(u_1, \ldots, u_r) \in M^- \rightarrow$
            Delete from $G$, all edges of the form $(u_i, x)$, $2 \leq i \leq r$;
            Delete from $G$, all edges of the form $(x, u_i)$, $1 \leq i \leq r - 1$;
            Delete from $G$ the edge $(u_r, u_1)$, if present;
            Collapse the path into a single vertex;
        rof;
    od;
    return $P$;
end;
```

Figure 2: Directed Matching Algorithm for LPP

THEOREM 3. *Let $(G = (V, E), \ell)$ be an instance of* LPP, *let M be a maximum directed matching of G and let k be the minimum number of edges in any cycle defined by M. $\lambda^*(G, \ell) \leq \frac{k}{k-1}\ell(M^-)$. In particular, $\lambda^*(G, \ell) \leq 2\ell(M^-)$.*

We note that $\delta^*(G, \ell) \geq \lambda^*(G, \ell)$. Hence, it provides a measure of how close a given solution is to optimal. We expect that the solutions obtained by breaking cycles will often be much closer to optimal than the bound in the theorem implies.

Theorem 3 provides the basis for our next approximation algorithm for LPP, shown in Figure 2. This algorithm constructs a maximum directed matching M in G, then breaks all the cycles in M and constructs a new graph in which the paths of M correspond to vertices. It then proceeds by finding a maximum directed matching in the new graph, continuing in this fashion until a Hamiltonian path in the original graph has been found.

Theorem 3 implies that $\psi^*(G, \ell) \leq 2\psi_{\text{DIMATCH}}(G, \ell)$ for any instance (G, ℓ) of LPP. The running time of DIMATCH is determined primarily by the directed matching algorithm used. Assuming an algorithm that runs in $O(n^{5/2})$ time, we get a running time of $O(n^{5/2} \log n)$ for DIMATCH.

DIMATCH is essentially an adaptation of an algorithm for the asymmetric traveling salesman problem (TSP) described by Karp in [7]. Karp's algorithm has poor worst-case performance for TSP, but performs well in a probabilistic sense for instances in which inter-city distances are selected uniformly on the interval [0, 1]. We have simply adapted his algorithm to the longest path problem (simplifying it slightly in the process), and observed that its worst-case performance is provably good in this context.

Greedy algorithm. The algorithms considered above are both fairly complicated and time consuming because they require the calculation of maximum weighted matchings. Another algorithm that is worth considering is the simple greedy algorithm that scans the edges in non-increasing order of length and selects an edge (u, v) if it has not previously selected an edge of the form (u, x) or (y, v) and if the collection of paths constructed so far does not include a path from v to u. The next theorem gives a worst-case bound on the performance of the greedy algorithm.

THEOREM 4. *If (G, ℓ) is an instance of* LPP *then $\lambda^*(G, \ell) \leq 3\lambda_{\text{PGREEDY}}(G, \ell)$.*

proof. Let F be the set of edges in some optimum solution to (G, ℓ). Let $H = \{h_1, \ldots, h_s\}$ be the set of edges chosen by the greedy algorithm in the order in which they were selected (that is, h_1 was selected first, h_2 second, and so forth).

We say an edge is *permissible* at some stage of the execution of the algorithm if its selection hasn't been precluded by earlier selections. Define H_i to be the set of edges which are permissible just before h_i is selected, but not permissible after h_i is selected.

Let $h_i = (u, v)$ and consider the situation just before h_i is selected by the greedy algorithm. At this point, u is the last vertex of some path constructed by the algorithm and v is the first vertex of

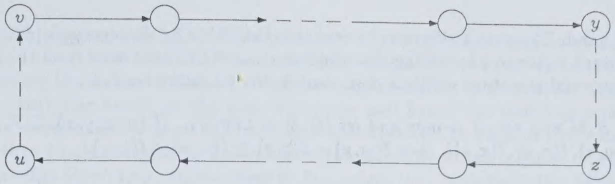


Figure 3: Illustration for Theorem 2.4

```
procedure PGREEDY(digraph $G = (V, E)$, edgelengths $\ell$,
        mapping left, right : $V \mapsto V \cup \{\text{null}\}$)
    vertex $u, v$; mapping leftend, rightend : $V \mapsto V$;
    for $u \in V \rightarrow$
        left(u), right(u) := null;
        leftend(u), rightend(u) := u;
    rof;
    Sort $E$ from longest to shortest;
    for $(u, v) \in E \rightarrow$
        if right(u) = null and left(v) = null and $v \neq$ leftend(u)
            right(u), left(v) := v, u;
            rightend(leftend(u)) := rightend(v);
            leftend(rightend(v)) := leftend(u);
        fi;
    rof;
end
```

Figure 4: Greedy Algorithm for LPP

some path (one or both paths may contain just a single vertex). Let z be the first vertex on the path containing u and let y be the last vertex on the path containing v as shown in Figure 3.

If there is an edge from y to z that is permissible before the selection of h_i then that edge is a member of H_i. All other members of H_i have the form (u, x) or the form (x, v).

Next, note that $\ell(h_i) = \max\{\ell(e) \mid e \in H_i\}$ and that $(H_1, \ldots, H_s)$ is a partition of E. Finally note that for $i \in [1, s]$, $|F \cap H_i| \leq 3$. This yields the theorem. □

Figure 4 is a sketch of an implementation of the greedy algorithm. Upon return, the mappings *left* and *right* give the left and right neighbors of each vertex in the solution path. If *left*(u) is null, then *rightend*(u) gives the vertex at the end of the path containing vertex u in the current partial solution. *leftend*(u) is similar. The running time for this implementation is $O(n^2 \log n)$.

3. A Greedy Algorithm for SCS

The greedy algorithm for the longest path problem can be restated for SCS as follows. Given a nonempty set of strings S, repeat the following step until S contains just one string.

> Select a pair of strings $s_1, s_2 \in S$ that maximizes $\psi(s_1, s_2)$. Remove s_1 and s_2 from S, replacing them with $s_1 \circ s_2$.

We refer to this algorithm as SGREEDY. Gallant [5] claims that $\phi_{\text{SGREEDY}}(S) \leq (3/2)\phi^*(S)$. This is not in fact true, as can be seen by considering the set of strings

$$S = \{\mathsf{abcbcbcbcb}, \mathsf{cbcbcbcbc}, \mathsf{bcbcbcbcbd}\}$$

for which $\phi_{\text{SGREEDY}}(S) = 20 > (3/2)\phi^*(S) = 19.5$. One can easily generalize this example to show that there is no constant $c < 2$ for which $\phi_{\text{SGREEDY}}(S) \leq c\phi^*(S)$. We currently do not know if there is some constant $c > 2$ for which $\phi_{\text{SGREEDY}}(S) \leq c\phi^*(S)$.

On the other hand, Theorem 4 allows us to conclude that $\psi^*(S) \le 3\psi_{\text{SGREEDY}}(S)$. In fact, we can improve the constant factor to 2 by noting that the instances of LPP that arise from the transformation from SCS have a special structure which is described in the following lemma.

LEMMA 1. *Let S be any set of strings and let $(G, \ell) = \text{LPP}(S)$. If $\{w, x, y, z\} \subseteq V$ with $\ell(w, y) = \max\{\ell(w, y), \ell(w, z), \ell(x, y), \ell(x, z)\}$ then $\ell(w, y) + \ell(x, z) \ge \ell(w, z) + \ell(x, y)$.*

proof. Identify w, x, y, z with the corresponding strings in S and let $\alpha = \ell(w, y)$, $\beta = \ell(x, z)$, $\gamma = \ell(w, z)$, $\delta = \ell(x, y)$. Note that if $\alpha \ge \gamma + \delta$ the result follows immediately. We will assume therefore that $\alpha \le \gamma + \delta$.

We define some notation for designating substrings. If $s = a_1 \ldots a_r$ is a string, $s[i]$ denotes the symbol a_i if $i > 0$ and a_{r+i+1} if $i < 0$. The notation $s[i, j]$ denotes the substring $s[i] \ldots s[j]$.

By definition of G,

$$\begin{aligned} w[-\alpha, -1] &= y[1, \alpha] && (1)\\ w[-\gamma, -1] &= z[1, \gamma] && (2)\\ x[-\delta, -1] &= y[1, \delta] && (3) \end{aligned}$$

From this we find

$$\begin{aligned} z[1, \gamma + \delta - \alpha] &= w[-\gamma, \delta - \alpha - 1] && \text{from (2) and } \alpha \le \gamma + \delta\\ &= y[\alpha - \gamma + 1, \delta] && \text{from (1)}\\ &= x[\alpha - \gamma - \delta, -1] && \text{from (3)} \end{aligned}$$

Hence, $\beta = \psi(x, z) \ge \gamma + \delta - \alpha$. □

THEOREM 5. *Let S be any set of strings. $\psi^*(S) \le 2\psi_{\text{SGREEDY}}(S)$.*

Proof. Let $(G, \ell) = \text{LPP}(S)$. Let $H = \{h_1, \ldots, h_s\}$ be the set of edges chosen by the greedy algorithm in the order in which they were selected (that is, h_1 was selected first, h_2 second, and so forth). Define λ_i^* to be the the length of a longest path set on the subgraph of G defined by the set of edges that are still permissible after the selection of h_i. We show that for $i \in [1, s]$, $\lambda_{i-1}^* \le 2\ell(h_i) + \lambda_i^*$. This in turn, implies the theorem.

Let $h_i = (w, y)$ and let X be a longest path in the subgraph of G defined by the set of edges that are permissible just before the selection of h_i (so $\ell(X) = \lambda_{i-1}^*$). By definition of the greedy algorithm, $\ell(w, y) = \max\{\ell(e) \mid e \in X\}$. At most three edges of X are not permissible after h_i is selected. If at most two become impermissible, then clearly $\lambda_i^* \ge \lambda_{i-1}^* - 2\ell(w, y)$ as desired.

If three edges become impermissible then one must have the form (w, z) with $z \ne y$, another the form (x, y) with $x \ne w$ and the third one, e joins the last vertex on the path containing y with the first vertex on the path containing w. This means that $X \cup \{h_1, \ldots, h_{i-1}\}$ contains a path from x to z, which means that (x, z) is permissible *after* (w, y) is selected. By Lemma 1, $\ell(w, y) + \ell(x, z) \ge \ell(w, z) + \ell(x, y)$. Consequently,

$$2\ell(w, y) + \ell(X \cup \{(x, z)\} - \{e, (x, y), (w, z)\}) \ge \ell(X)$$

which implies $\lambda_i^* \ge \lambda_{i-1}^* - 2\ell(w, y)$. □

A naive implementation of SGREEDY takes at least quadratic time. A similar running time is obtained if one uses the transformation to LPP and then uses PGREEDY. A much faster running time can be obtained however by making use of an appropriate data structure. For our purposes, a *suffix tree* T is an abstract data type representing a collection of strings $S = \{s_1, \ldots, s_n\}$ on which the following operations are defined.

T.SUFFIX_TREE($S = \{s_1, \ldots, s_n\}$)	Initialize T to represent the strings in S. This operation may only be performed once.
T.LOOKUP(integer i, j)	Returns a pair $[\ell, k]$, where ℓ is the length of the longest prefix of s_i which is also a suffix of some string in $S - \{s_j\}$ and s_k is one such string.
T.DELETE(integer i)	Removes s_i from the set of strings represented.

The obvious implementation of a suffix tree is a *trie* (see [1]) containing an entry for every suffix of every string in the set. We assume that in addition to the edge labels, the nodes contain lists of integers, identifying the strings in the set that have suffixes ending at that node. This representation doesn't quite satisfy our needs, as the size of the trie and hence the time required to construct it is $\Omega(m^2)$ in the worst-case. A more compact representation can be obtained by labeling edges with strings rather than single characters. This yields a representation that requires $O(m)$ space and that can be constructed in $O(m)$ time, as described by McCreight [11]. See also [2,12]. (Actually, McCreight defines a suffix tree to contain suffixes of a single string rather than a collection of strings. Our variant requires only minor modifications to McCreight's method.)

We perform deletion in suffix trees using lazy deletion. That is, to delete a string s_i, we simply mark it deleted in an auxiliary bit vector maintained for this purpose. When a lookup operation is performed, we probe the tree to find the longest match. Let u be the node at which the probe terminates. The list of matching strings at u is scanned and any that are marked deleted are removed from the list. If this makes the list empty and u has no children, then u is removed from the tree. If no acceptable match can be found in the list, the search continues at the parent of u.

The time required for a single lookup operation may well exceed the length of the string being searched for. However, any excess time is spent deleting node list entries. Since there are initially $m+n$ such entries in the whole tree, the time spent on any sequence of lookups is $O(m)$ plus the sum of the lengths of all strings being searched for.

We say that a sequence of lookup and delete operations is *monotonic* if for every i, j, k with $j \neq k$, whenever the sequence contains the operation T.LOOKUP(i, j) and later on the operation T.LOOKUP(i, k), it contains the operation T.DELETE(j) between the other two.

We can speed up a monotonic sequence of operations, by maintaining for each string, a pointer to the node where the most recent lookup for that string ended. This allows us to avoid the initial probe of the tree when we perform a lookup operation. Instead, we use the pointer to go straight to the node where the last probe ended, and search up from that node if necessary. In this way, we can perform a monotonic sequence of r operations in $O(m+r)$ time.

This analysis assumes that the symbol alphabet is small enough that it is reasonable to use a vector of pointers to children in each node, indexed by the first symbol of the strings labeling the edges. If a large alphabet is needed, a hash table may be used. Another option is to use a variant on Sleator and Tarjan's *lexicographic splay tree* [14]. With this representation, the time required to perform a sequence of operations is $O(m)$, plus the sum of the lengths of the strings being searched for, plus $O(\log m)$ per operation. For a monotonic sequence of r operations the time is $O(m + r \log m)$.

An efficient implementation of the greedy algorithm for strings is shown in Figure 5. The algorithm does not explicitly combine strings, but keeps track of the decisions made using the two mappings *left*(i) and *right*(i), which give the left and right neighbors of string i in the solution constructed so far. A value of 0 means that there is no neighbor. The solution is returned in these mappings. If a string i has no left neighbor yet, *rightend*(i) is the original string which is currently rightmost in the piece of the partial solution that contains i; *leftend*(j) is similar.

The heap h, is used to determine which pair of strings should be combined next. Each string is entered in h with the *key* being the length of the best match for h. As the algorithm proceeds, certain matches become unavailable and the values of *key*() may become invalid. Consequently, whenever a string s_i is selected from h, a new lookup operation is performed in T. If the result of that operation is a match of the same length as *key*(i), the strings are combined. If the lookup results in a shorter match, the value of *key*(i) is changed and the position of s_i in the heap is adjusted to reflect the new value. Note that a string is deleted from the heap once it is successfully matched with another string on its left end. Similarly, a string is deleted from the suffix tree once it is matched with a string on the right.

The running time of the algorithm is dominated by the operations on the data structures within the main loop. The number of iterations of the main loop is $O(m)$ in the worst case. Since the heap operations are $O(\log n)$ per operation, using the usual implementation, the total time spent on the heap operations is $O(m \log n)$. This can be improved to $O(m + n \log n)$ by using Fiboncacci heaps [3]. Since the sequence of operations on the suffix tree is monotonic, the time needed for the suffix tree operations is $O(m)$ assuming a small alphabet and $O(m \log m)$ assuming a large alphabet and the use of lexicographic splay trees. This yields a total running time of $O(m + n \log n)$ for small alphabets and $O(m \log m)$ for large alphabets.

```
procedure SGREEDY(string_set $S = \{s_1, \ldots, s_n\}$, mapping $left, right : [1,n] \mapsto [0,n]$)
    integer $i, j, \ell$; mapping $key : [1,n] \mapsto$ integer ;
    mapping $leftend, rightend : [1,n] \mapsto [1,n]$;
    heap $h$; suffix_tree $T$;
    $T$.SUFFIX_TREE($S$);
    for $i \in [1,n] \rightarrow$
        $left(i), right(i) := 0$;
        $leftend(i), rightend(i) := i$;
        $[key(i), j] := T$.LOOKUP($i, i$);
        $h$.INSERT($i$);
    rof;
    do $|h| > 1 \rightarrow$
        $i := h$.FINDMAX();
        $[\ell, j] := T$.LOOKUP($i, rightend(i)$);
        if $\ell = key(i) \rightarrow$
            $left(i), right(j) := j, i$;
            $leftend(rightend(i)) := leftend(j)$;
            $rightend(leftend(j)) := rightend(i)$;
            $T$.DELETE($j$); $h$.DELETE($i$);
        | $\ell < key(i) \rightarrow$
            $key(i) := \ell$; $h$.SIFTDOWN($i$);
        fi;
    od;
end
```

Figure 5: Greedy Algorithm for SCS

4. SCS and the Traveling Salesman Problem

In this section we relate SCS to the path version of the *traveling salesman problem* (TSP). An instance of the traveling salesman problem is a list of cities $C = (c_1, \ldots, c_n)$ with a distance $d(c_i, c_j)$ between each pair of cities. The object is to find a permutation π on $\{1, \ldots, n\}$. that minimizes

$$\theta_\pi(C, d) = \sum_{i=1}^{n-1} d(c_{\pi_i}, c_{\pi_{i+1}})$$

We define $\theta^*(C, d) = \min_\pi \theta_\pi(C, d)$.

Let $S = \{s_1, \ldots, s_n\}$ be an instance of SCS. We define TSP$(s_1, \ldots, s_n)$ to be an instance (C, d) of TSP with $C = (c_1, \ldots, c_n, c_{n+1})$ and

$$d(c_i, c_j) = \begin{cases} |s_i| - \psi(s_i, s_j) & 1 \le i \le n \quad 1 \le j \le n \quad i \ne j \\ |s_i| & 1 \le i \le n \quad j = n+1 \\ \|S\| & i = n+1 \quad 1 \le j \le n \end{cases}$$

Note that in general, if π satisfies $\theta_\pi(C, d) = \theta^*(C, d)$ then $\pi_{n+1} = n + 1$.

THEOREM 6. *Let $S = \{s_1, \ldots, s_n\}$ be an instance of* SCS, *$(C, d) =$* TSP*(S) and let π be a permutation on $\{1, \ldots, n, n+1\}$ for which $\pi_{n+1} = n + 1$. Then $\theta_\pi(C, d) = \phi_{\pi'}(S)$, where π' is the restriction of π to $\{1, \ldots, n\}$. In particular, $\theta^*(C, d) = \phi^*(S)$.*

Theorem 6 implies that any good approximation algorithm for this version of the traveling salesman problem is a good approximation algorithm for SCS as well. The particular instances of TSP constructed by the transformation defined above have some special properties. First, they may be asymmetric; that is $d(c_i, c_j)$ need not equal $d(c_j, c_i)$. The next theorem shows that they obey the so-called *triangle inequality*.

THEOREM 7. *Let $S = \{s_1, \ldots, s_n\}$ be an instance of* SCS *and let $(C, d) =$* TSP*(S). For all $c_i, c_j, c_k \in C$, $d(c_i, c_k) \le d(c_i, c_j) + d(c_j, c_k)$.*

There exist efficient approximation algorithms for the symmetric version of the traveling salesman problem with triangle inequality that produce solutions within a factor of (3/2) of optimal. When the triangle inequality doesn't hold, finding good approximate solution is as difficult as finding optimum solutions (see [8]). For the asymmetric version with triangle inequality however, little is known. There are no known approximation algorithms that are both efficient and have good worst-case performance (nor have we found any), but the approximation problem has not been shown to be hard. Consequently, the relationship between SCS and TSP has yet to yield any immediately useful results. The relationship does imply some consequences if the status of either problem is resolved in the future. If good approximation algorithms are found for TSP, they may be applied to SCS. If the approximation problem for SCS is shown to be hard, then the approximation problem for TSP must be hard. It also may be that good approximation algorithms discovered for SCS, could be adapted to TSP, although this does not necessarily follow. Finally, a proof that TSP is hard to approximate would not imply that SCS is hard to approximate, but it would imply that approximation algorithms for SCS would have to exploit special structural properties present in the instances of TSP that result from our transformation. We close by mentioning one such property that may prove useful.

LEMMA 2. *Let S be any set of strings and let $(C, d) = \text{TSP}(S)$. If $\{w, x, y, z\} \subset C$ with $d(w, y) = \min\{d(w, y), d(w, z), d(x, y), d(x, z)\}$ then $d(w, y) + d(x, z) \leq d(w, z) + d(x, y)$.*

We refer to this as the *quadrilateral inequality* and note that it is analogous to the property described in Lemma 1.

5. Closing Remarks

We have related the shortest common matching string problem to two other problems, using transformations that preserve solution values. These two transformations reflect different ways of viewing SCS. We have used the transformations to gain insight into the problem of approximating SCS and have discovered several algorithms that have provably good performance with respect to the overlap measure. The best of these is the string version of the greedy algorithm for which we have described an efficient implementation using suffix trees.

While we have shown that the string version of the greedy algorithm has good worst-case performance with respect to the overlap measure, we cannot determine its performance with respect to the length measure. We know that it can be off by as much as a factor of two with respect to the length measure, but we don't know if it can be worse than this. One open problem then, is to resolve this issue.

Although, we have been unable to make use of the relationship between SCS and TSP to advantage, we feel that it may yet prove useful. More generally, we think that the use of transformations that preserve solution values can be used to extend the application of known approximation algorithms to new domains. A methodical development of such transformations could provide many useful results.

Another worthwhile line of investigation is to study the probable performance of the various approximation algorithms using appropriate probability models. It appears likely for example, that the directed matching algorithm for the longest path problem performs much better than its worst-case bound would indicate for a wide class of natural probability models. Similarly, one would expect the greedy algorithm to perform well in a probabilistic sense for many useful probability models.

References

[1] Aho, Alfred V., John E. Hopcroft and Jeffery D. Ullman. "Data Structures and Algorithms," Addison-Wesley, 1982.

[2] Chen, M. T. and J. I. Seiferas. "Efficient and Elegant Subword-Tree Construction," University of Rochester, Computer Science Department, technical report TR 129, 12/83.

[3] Fredman, Michael L. and Robert E. Tarjan. "Fibonacci Heaps and Thier Uses in Improved Network Optimization Algorithms," *Proceedings of the IEEE Symposium on the Foundations of Computer Science*, 10/84, 338–346.

[4] Gallant, John K., David Maier, James A. Storer. "On Finding Minimal Length Superstrings," *Journal of Computer and System Sciences*, vol. 20, no. 1, 50–58, 2/80.

[5] Gallant, John K. "String Compression Algorithms," Ph.D. Dissertation, Princeton University, Department of Electrical Engineering and Computer Science, June 1982.

[6] Gingeras, T. R., J. P. Milao, D. Sciaky and R. J. Roberts. "Computer Programs for the Assembly of DNA Sequences," *Nucleic Acids Research*, vol. 7, 1979, 529–545.

[7] Karp, Richard M. "A Patching Algorithm for the Nonsymmetric Traveling Salesman Problem," *SIAM Journal on Computing*, vol. 8, no. 4, 561–573, 11/79.

[8] Lawler, E. L., J. K. Lenstra and A. H. G. Rinnooy Kan (editors). "The Traveling Salesman Problem," John Wiley and Sons, Ltd, 1986.

[9] Maier, David and James A. Storer. "A Note on the Complexity of the Superstring Problem," Princeton University Technical Report 233, Department of Electrical Engineering and Computer Science, October 1977.

[10] Mayne, A. and E. B. James. "Information Compression by Factorising Common Strings," *The Computer Journal* 18, 1975, 157–160.

[11] McCreight, Edward M. "A Space-Economical Suffix Tree Construction Algorithm," *Journal of the ACM*, vol. 23, 4/76, 262–272.

[12] Rodeh, M., V. R. Pratt and S. Even. "Linear Algorithm for Data Compression via String Matching," *Journal of the ACM*, vol. 28, 1/81, 16–24.

[13] Shapiro, M. B. "An Algorithm for Reconstructing Protein and RNA Sequences," *Journal of the ACM*, vol. 14, 1967, 720–731.

[14] Sleator, Daniel D. and Robert E. Tarjan. "Self-Adjusting Binary Search Trees," *Journal of the ACM*, vol. 32, 7/85, 652–686.

[15] Stefik, M. "Inferring DNA Structures From Segmentation Data," *Artificial Intelligence*, vol. 11, 1978, 85–114.

[16] Storer, James A. and Thomas G. Szymanski. "Data Compression via Textual Substitution," *Journal of the ACM*, vol. 29, 10/82, 928–951.

[17] Tarjan, Robert E. "Data Structures and Network Algorithms," Society for Industrial and Applied Mathematics, 1983.

AN EXPERIMENTAL STUDY OF MEDIAN-SELECTION IN QUICKSORT

CATHERINE COLE McGEOCH

Carnegie-Mellon University, Pittsburgh, PA 15213

Abstract

A well-known method for choosing the partition element during recursive stages of Quicksort is to use the median of a random sample of size T selected from the subarray. The best choice of T reflects a tradeoff between the cost of finding the sample median and the quality of the partiton. This paper presents an experimental study of this tradeoff for fixed T as well as for sample sizes that vary as a function of subarray size.

1. Introduction

Quicksort is among the most efficient of comparison-based sorting methods. Proposed by Hoare [2, 3] in 1960, the algorithm was thoroughly analyzed by Sedgewick [8, 7], who gave precise performance bounds for a number of implementation strategies. Knuth [5] also provides a detailed discussion.

To sort an array X of N elements, Quicksort partitions X around a *partition element*, s, so that elements with value less than s are to its left in the array and larger elements are to its right; after partitioning, s is in its correct position. The algorithm then recurs on the two subarrays on either side of s. Performance depends to a great extent upon the choice of partition element at each stage of the recursion. If, for example, the least element is selected each time, then Quicksort can require $\Omega(N^2)$ total comparisons during partitioning. Best performance is is achieved if the list is divided in half at each stage. A number of efficient strategies for selecting a partition element with rank near the median have been examined.

A strategy that works well is to choose s at random at each stage. A generalization (suggested by Hoare) takes a sample of size T from the sublist and uses the median of the sample as an estimate of the true median (random selection corresponds to $T = 1$). Singleton [9] and Sedgewick [8] recommended *median-of-3* Quicksort (that is, $T = 3$) over a large class of selection strategies. Sedgewick showed that the percentage improvement in sorting cost is small for larger T and argued that the cost of finding the median of larger samples would quickly overtake any improvement in sorting cost. His argument was not made precise because the comparisons used in median selection were not an explicit part of his analytical model; instead, this cost was included as part of the overhead for each recursive call.

This paper extends Sedgewick's analysis by explicitly including the cost of median selection and by considering the trade-offs between partitioning and median-selection costs. Intuitively, larger sample sizes require more median-selection comparisons but produce better partitions, giving fewer partition comparisons. The following section discusses the simulation model and implementation issues. Section 3 presents simulation results that compare fixed-T strategies; although comparison cost is of primary interest, others measures are also considered. Section 4 examines strategies that allow T to vary as a function of the sublist size at each recursive stage. Section 5 corrects a small error in Sedgewick's analysis of median-of-3 Quicksort.

2. Simulation Issues

It is not necessary to generate and sort random lists of numbers to obtain the desired measurements. A simple "shortcut" simulation program mimics the performance of Quicksort on random lists with much less computation time. This section describes the model of Quicksort to be simulated as well as details of the simulation program.

Assume in the following that the N elements to be sorted consist of the integers 1 through N, initially arranged in some random permutation; this simplifies the discussion as well as the simulation since the rank of an element is identical to its value. Note that this implies that equal-valued elements do not occur in the input list; otherwise, this assumption does not affect the analysis of Quicksort because performance depends only on the ranks and not on the values of the elements. (See Sedgewick [8] for a discussion of Quicksort with equal elements). The general structure of Quicksort is well known. The basic model can be improved by not recurring on sublists smaller than some cutoff M. to sort the elements in the small sublists, a single final pass of Insertion Sort, which has low overhead, is used.

Quicksort and Insertion Sort are among the few algorithms for which implementation strategies have been analyzed exactly rather than in asymptotic order-of-magnitude terms. To do this, a set of measures are identified that correspond to the number of times various pieces of code are executed. The analysis remains independent of specific implementation because the measures correspond to the *number* of times various steps are performed rather than to their running times. The standard model incorporates the measures listed below; specific values at each recursive stage correspond to Sedgewick's very efficient implementation of Quicksort [8, 7].

- A: the expected number of times subroutine **Qsort** is called. At each recursive stage A is incremented by 1.
- B: the expected total number of *exchanges* performed during partitioning. Equivalently, the number of elements that must be moved in order to partition the subarray around the partition element **s**. If the array is of size $n =$ **hi - lo + 1** and T is the sample size, then at each stage B the expected number of exchanges is given by $(n-1)(n-s)/\binom{n}{T}$.
- C: the expected total number of *comparisons* of array elements to **s** during partitioning (but not during median-selection). In Sedgewick's implementation C is incremented by $n-1$ at each recursive stage. This implementation requires a high-valued sentinel at the high end of array **X**. subarrays.
- D: the number of *insertions* performed during Insertion Sort. D represents the sum, over the subarrays of size less than M, of the number of insertions in each. For each small subarray of size $n \leq M$ the number of insertions has expected value $n - H_n$, where H_n denotes the n^{th} harmonic number.
- E: the expected number of *moves during insertion* (equivalently, the total distance items are shifted) performed by Insertion Sort. For each small subarray of size $n \leq M$, this has expected value $n(n-1)/4$. Like D, E represents the sum of this quantity over all small subarrays.

Let $t(n)$ be a function that returns an odd-valued integer in the range $[1, n]$: then $T = t(n)$ is the size of the sample taken from the n elements at a given recursive stage. Since the sample size is allowed to vary as a function of the subarray size at each stage, a general median-selection algorithm that takes the sample size as input is required. In Sedgewick's very fast implementation the sample elements are evenly-spaced over the subarray. General median-selection algorithms assume a *contiguous* set of elements, rather than an evenly-spaced sample of a larger set: how shall median-selection be embedded into Quicksort? Detailed evaluation of implementation possibilities would depend upon many factors specific to the environment; this (interesting) problem is more appropriate to a study that would measure program performance on a specific machine. The policy adopted for this simulation study was to keep it simple, to make as few assumptions as possible about implementation details, and to allow for the possibility of in-place median selection.

The selection method used for the simulation is the linear-expected time algorithm first described by Hoare [2]. Let $f_{T,k}$ represent the number of *selection comparisons* required to find the k^{th} largest of T elements, and let $g_{T,k}$ represent the number of *selection exchanges*. Knuth

[6] solved $f_{T,k}$, showing that $f_{T,k} = 2[(T+1)H_T-(k+1)H_k-(T-k+2)H_{T-k+1}+T+5/3]$. It appears that a similar closed form for $g_{T,k}$ has not been published. Let F represent the *total* number of comparisons performed during median selection.

When T is fixed and $M \leq T$, it is possible that a subarray is too small: how does one sample 5 elements from 3? Sedgewick's implementation samples *with replacement* in such a case, so that some of the elements are duplicated. Because this assumption is not compatible with that of sampling and selecting from contiguous elements the subarray, for this study the elements are sampled without replacement. Instead, when T is greater than the subarray size n, T is set to n or $n-1$, whichever is odd. This implies that the true median is *always* found when n is odd and smaller than T, which may not be the case under Sedgewick's model. If T is less than the cutoff M, then this model is identical to Sedgewick's.

An obvious simulation strategy is to generate random lists of numbers and to sort them while recording the measures of interest. This would require $\Omega(N)$ steps to generate each list plus $\Omega(N \log N)$ steps to sort. A simple observation allows more efficient simulation; the "shortcut" implementation described below is similar to a shortcut Bentley describes for a median-selection algorithm in [1].

Rather than selecting the partition element from the subarray at each recursive stage, the shortcut simulation program generates a partition element from a hypothetical subarray according to an appropriate probability distribution. At each stage, the *size of the subarray* (equal to **hi** - **lo** + 1) and the *value of the partition element* (randomly generated by a procedure to be described shortly) are the only two quantities needed to accumulate the measures described earlier. Note that the *expected* values of B, D, E, and F are accumulated at each level, rather than values of corresponding random variables. As a result, the variance for these four quantities is much smaller than would be displayed by Quicksort. Since only means are examined here, this "bug" becomes a feature: the small variance in experimental results means that few trials are needed. This is an application of a standard variance reduction technique known as Conditional Monte Carlo.

The generation of an appropriate partition element is easily accomplished in $O(t(n))$ time at each stage: simply generate a random sample of size T from $[1, n]$ and return the sample median. Two methods are employed in the simulation program, the choice depending on whether T is near n. When $T \ll n$, the simulation program generates integers uniformly T distinct ones appear; a hash table of size $2T$ is used to check for duplicates. When T is near n this method is inefficient because of the large number of duplicates generated. Knuth [4] (Section 3.4.2) gives an algorithm for generating a sample of T integers from $1..n$ in *ascending order* by considering each integer in turn and "accepting" it with appropriate probability. This algorithm is modified in the simulation program to stop when the m^{th} integer is accepted. This process requires about $T/2$ random number calls; as implemented, it is more efficient than the first method when $T \leq \sim 0.3n$. Note that because the elements are maintained in sorted order (making it easier to find the median), both of these methods produce partition elements more efficiently than if median-selection had actually been performed during Quicksort. The second method is even more efficient because it only considers approximately $T/2$ elements.

The simulation program is sketched in Exhibit 2-1; the formulas for accumulating A through F reflect the simulation model described earlier. The running time of the simulation program depends upon $t(n)$. When $t(n)$ is a constant function, the program runs in time linear in N.

The experiments described in the following sections were performed on a VAX 11/780 running under Unix[1]. The random number generator was the cyclic feedback method described by Knuth

---

[1] VAX is a trademark of Digital Electric Corporation. Unix is a trademark of AT&T Bell Laboratories.

```
Procedure Shortcut(n)
    if (n < M) then
        D += n - $H_n$                  Accumulate Insertion Sort measures.
        E += n(n - 1)/4
    else
        T = t(n)                        Determine sample size.
        m = (T+1)/2
        s = Generate-Partition-Element(n, T)
        A += 1                          Accumulate measurements.
        B += (n-s)(s-1) / Choose(n, T)
        C += n - 1
        F = 2((n + 1)$H_n$ - (m+1)$H_m$ - (T-m+2)$H_{T-m+1}$+T+5/3

        Shortcut(s-1)                   Recur.
        Shortcut(n-s-1)
```

Exhibit 2-1: The Simulation Program

[4] (Algorithm A, Section 3.2.2, 2nd Edition). In most of the following, N is set at powers of two from 2^8 to 2^{18}, M is set at 0, 10, and 20, and T (in Section 3) is set at 1, 3, and 5.

To check the random number generator as well as existing formulas (see Section 5), a dynamic program was implemented to generate exact values for some of the measures for $T = 1$, 3, 5, and 7. In all tests, observed means for the measures were within 1.5% of the true means produced by the dynamic program.

3. Fixed-T Strategies

This section presents simulation results for median-of-T Quicksort for T fixed at 1, 3, and 5. A sample point is determined by N, M, and T. Most of the simulations were performed at sample points corresponding to $N = 2^8, 2^{10}, \ldots 2^{18}$, $M = 1, 10, 20$, and $T = 1, 3, 5$. Measures A through F are considered. An obvious problem in describing simulation results arises: how to represent a function of three variables using two-dimensional graphs?

A number of approaches to this problem are possible; for this study, the following conventions are adopted. Measurements for the three M settings appear in separate panels. In each panel the x-coordinate of a point is determined by $\log_2 N + T/10$, and the y-coordinate of each point corresponds to the specified measure (giving the mean over 20 trials at this sample point). For example, in the left panel of Graph 3-1-a, sample point $M = 1$, $N = 2^8 = 256$, $T = 1$ is represented by the leftmost cross in the panel. The second-leftmost point corresponds to the sample point $M = 1$, $N = 2^8$, $T = 3$, and is plotted with with x-coordinate 8.3. This method of "coding" the x-coordinate of each point allows easy comparison of the measures at in terms of the simulation parameters.

Let C' represent the average *total* number of comparisons required in fixed-T Quicksort; that is, $C' = C + F$. Graph 3-1-a depicts C'/N for the sample points given above. Not surprisingly, large M (right panel) gives fewer comparisons at every sample point since the cost of Insertion Sort is ignored. The reduction in Quicksort comparisons at high M must of course be balanced against the increase in Insertion Sort time. This would be an important task in determining the best choice of M for a specific implementation.

The left panel in Graph 3-1-a corresponds to $M = 1$; that is, to Quicksorting the entire array and not performing Insertion Sort at all. In this panel, the leftmost in each triple of crosses has least

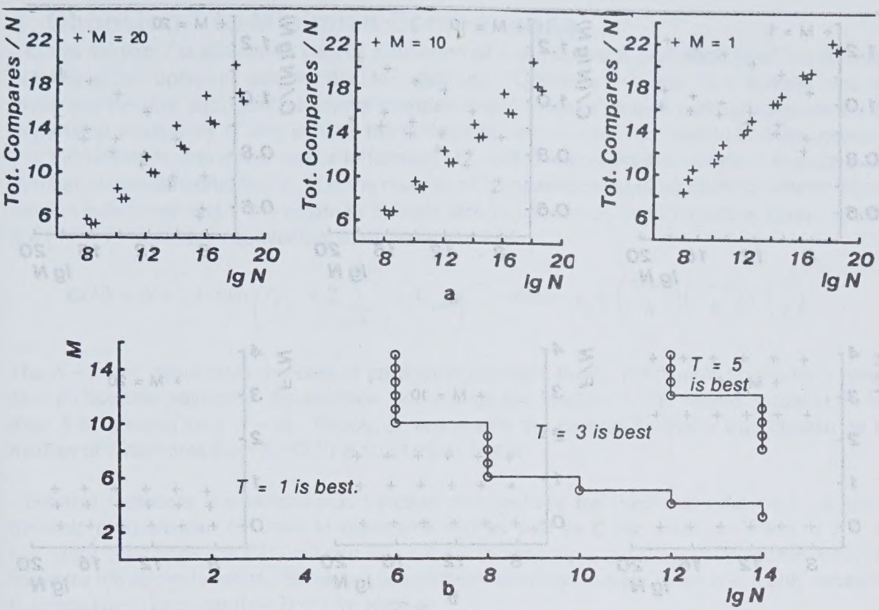


Exhibit 3-1: Total Comparisons

value when $N \le 2^{14}$, and the middle cross has least value when $N > 2^{14}$. This indicates that the Median-of-1 strategy gives fewest total comparisons when N is small, but Median-of-3 is best when N is large (within the range of the experiments). If a cutoff of size $M = 10$ is used (middle panel), then Median-of-3 gives fewest total comparisons for N smaller than 2^{16}, and Median-of-5 gives fewest total comparisons at higher N. If $M = 20$ (right panel), then Median-of-5 is best for all but the lowest value of N sampled. Note that the separation in cost between Median-of-1 and the other two strategies becomes more pronounced at large M.

Graph 3-1-b presents results of further simulations to determine the best choice of T for various combinations of N and M. Each region of the graph corresponds to the T setting that gives the smallest mean value of C' for N at even powers of 2 and M below 16, 20 trials each. For example, when either M or N is low (bottom left region), Median-of-1 Quicksort gives fewest total comparisons. Note that the x-scale corresponds to quadrupling N each time whereas the ordinate corresponds to unit increases in M.

Consider how C' is divided between C and F. Graph 3-2-a presents $C/(N \log N)$. Not surprisingly, C decreases as M grows, since the cost of Insertion Sorting small subarrays is ignored. Although the logarithmic x-scale in each panel makes the curves appear to grow more steeply than they actually do, the asymptotic constant is not easily seen at these values of N. As predicted, the triples of crosses show that C decreases in T: larger sample sizes tend to give better partitions, decreasing the total number of partition comparisons. Graph 3-2-a shows that for any combination of value of M and N, the $T = 1$ strategy (the leftmost symbol in each triple) gives a significantly higher value of C than $T = 3$ or $T = 5$. Although Median-of-5 gives the lowest value for C everywhere, the improvement over Median-of-3 is never very great. This agrees with Sedgewick's observation that while Median-of-3 gives a substantial improvement over Median-of-1 for this measure, the percentage improvement at higher values of T is small.

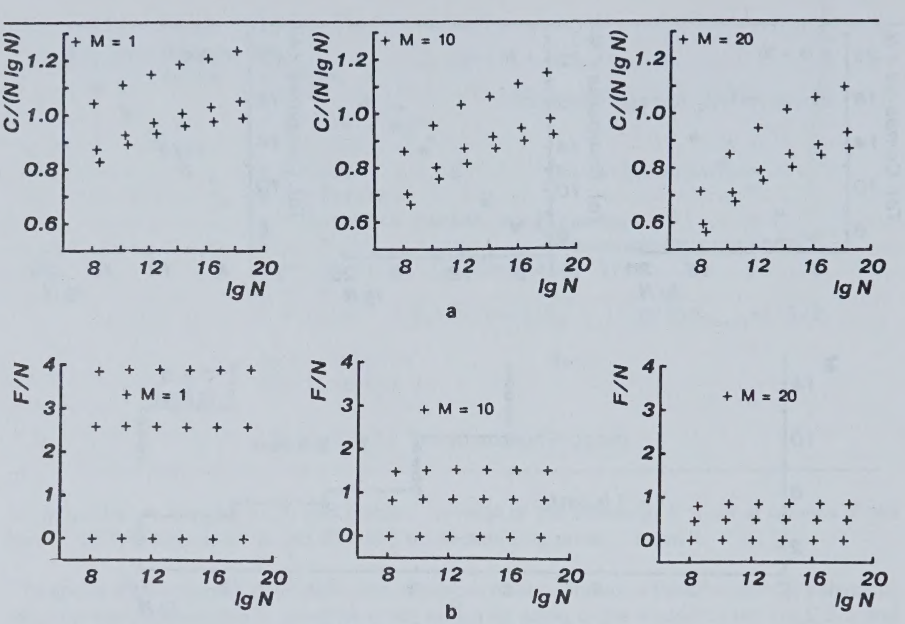


Exhibit 3-2: Measures C and F

For fixed T the total number of median-selection comparisons, F, must be proportional to the number of times the median selection routine is performed. Therefore, F must be proportional to A, the number of recursive stages seen during Quicksort. Specifically, for $m=(T+1)/2$, $F=f_{T,m}(T)\cdot A$, where $f_{T,m}(T)$ was given earlier by formula (1). Since the number of recursive stages reached is linear in N, F is also linear in N. Graph 3-2-b presents F/N. When $T=1$ no median-selection is performed, so the leftmost cross in each triple is always equal to 0. As predicted, F tends to increase in T. In both 3-2-a and 3-2-b the difference between $T=1$ and $T=3$ is more pronounced than the difference between $T=3$ and $T=5$, although one measure increases in T and the other decreases in T.

Suppose that the choice of T to minimize total comparisons at a given recursive stage is an increasing unbounded function of subarray size. If the best choice of T grows fast enough in n, then as N (the problem size) increases, the large subarrays would eventually overcome the small subarrays in "voting" for the best choice of T. When M is greater than 1 the very small subarrays are ignored, so the influence of the large subarrays is seen earlier in N. This would explain the observed behavior. An implication of this argument is that the choice of T to minimize total comparisons (over the set of fixed T strategies) does not have a constant upper bound, but rather increases with N (and the rate of increase is determined by M), contradicting Sedgewick's argument that Median-of-3 is the best choice among fixed-T strategies when $M=1$. (On the other hand, Graph 3-1-b suggests that Median-of-3 (or Median-of-1) is indeed the best choice over a large range of practical input sizes.) Another implication is that a strategy which varies the sample size at each level according to the subarray size would give fewer total comparisons than any fixed-T strategy. Evidence that this is the case is presented in Section 4.

Although space does not permit a detailed discussion, the behavior of the other measures as T and M vary is also of interest. Simulations reveal, for example, that the measure A is not monotonic in T: under certain conditions, large T will give *more* partitioning stages than small T.

4. Choosing T to Minimize Comparisons

In this section T is allowed to vary as a function of n, the sublist size at each level; the goal is to determine the optimum sample size for each n. "Optimum" means "the sample size that minimizes the total expected number of comparisons." Recall that $n-1$ comparisons are needed to partition a subarray of size n. Also, the number of comparisons required to find the median of $T = t(n)$ elements was given earlier in formula (1). Let C represent the *minimum* expected total number of comparisons; that is, C is the number of comparisons required (during partitioning and median selection) when the optimum sample size is chosen at each recursive stage. Letting $h = \lfloor T/2 \rfloor$ for notational convenience, we have for $> M$

$$\mathrm{C}(N) = N-1 + \min_{T}\left(f_{T,m} + 2\sum_{s=h+1}^{N-h} p_s \cdot \mathrm{C}_{s-1}\right) \qquad \text{where } p_s = \binom{N-s}{h}\binom{s-1}{h} \Big/ \binom{N}{T}.$$

The $N-1$ term represents the cost of partitioning at each level. The summation index s ranges over all possible values for the partition element (as the median of T elements, s cannot be less than h or greater than $N-h$). Finally, p_s represents the probability that s was chosen as the median of T elements from N. $\mathrm{C}(N) = \mathrm{O}(1)$ when $N \le M$.

Let $\mathrm{t}(n)$ represent the sample-size function that realizes the minimum total cost. A simple dynamic program can be used to determine $\mathrm{t}(n)$ as well as C for small n. When $M = 1$, the boundary conditions are $\mathrm{C}(1) = 0$ and $\mathrm{t}(1) = 0$; for increasing n, the program searches for $\mathrm{t}(n)$ to minimize the above function. Since $\mathrm{t}(n)$ is nondecreasing and at most linear in n, it only necessary to check $\mathrm{t}(n-1)$ against $\mathrm{t}(n-1)+2$ for each n.

Least N	t(·)
1	1
35	3
93	5
197	7
337	9
515	11
730	13
984	15
1274	17
1603	19
1968	21
2372	23
2813	25

a

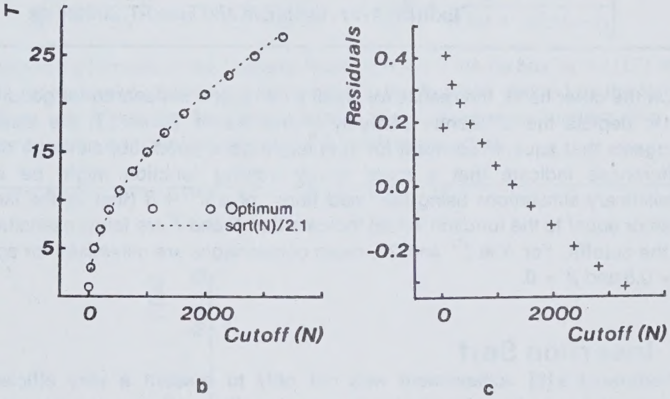


Exhibit 4-1: Cutoffs for t(n)

Table 4-1-a gives the lower boundary value of n corresponding to each $\mathrm{t}(n)$, for $n \le 3000$. The second and third rows indicate, for example, that 3 is the optimum choice of T for subarrays of size between 35 and 92. Graph 4-1-b presents the table entries in graphical form; the dotted line corresponds to the function $N^{1/2}/2.1$. The residuals, representing the difference between the optimal values and this function, are presented in Graph 4-1-c. Similar computations for $M = 10$ differ from Table 4-1-a in that $\mathrm{t}(3)$ has a lower bound of 30 rather than 35; otherwise the table entries are identical.

Exhibit 4-2 compares C to C' (corresponding to the fixed-T strategies), for the four sample points $N = 256, 1024$ and $M = 1, 10$. In each triple, the circles gives the ratio C'/C for $T = 1, 3, 5$, respectively. As predicted in the previous section, the strategy that modifies sample size according to subarray size gives fewer total comparisons than any of the fixed-T strategies at these four sample points. While these results are encouraging, it is difficult to determine $\mathbf{t}(n)$ at higher n. When $N = 2500$, for example, the dynamic program must compute the combination "2500 choose 24": although careful programming could push the computation higher, machine precision becomes a significant factor. Solving the problem analytically also seems to be difficult. A solution to the recursion for C with arbitrary function $t(n)$ is required in order to find the $t(n)$ that minimizes C.

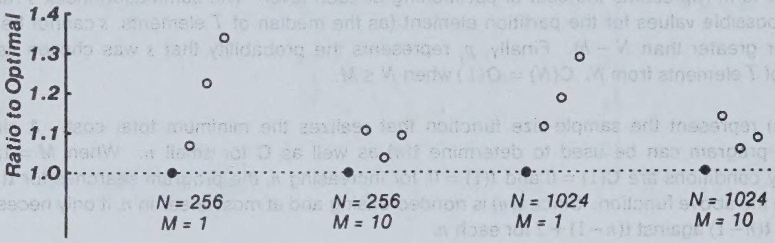


Exhibit 4-2: Optimum and Fixed-T Strategies

On the other hand, the results for small n can guide the search for good functions $t(n)$. Graph 4-1-c depicts the difference between t_n and the fit $Y = n^5/2.1$; the small magnitude of error suggests that square-root form for $t(n)$ might do a good job, although the steady decrease in differences indicate that a more slowly-growing function might be needed at higher N. Preliminary simulations using the "odd floor" of $\alpha N^{1/2} + \beta$ (that is, the largest odd integer less than or equal to the function value) indicate that C and F are fairly insensitive to small differences in the cutoffs. For N at 2^{12} and 2^{16}, mean comparisons are minimized for square-root forms when $\alpha = 0.5$ and $\beta = 0$.

5. Insertion Sort

Sedgewick's [8] achievement was not only to present a very efficient implementation of Quicksort, but to demonstrate that it was more efficient than many alternatives by deriving exact formulas for quantities A through E. His final version employs such techniques as removal of tail-recursion, loop unrolling, careful ordering of conditionals, and fine-tuning of parameters. It uses a Median-of-3 selection strategy with a cutoff for Insertion Sort at $M = 9$.

One of my first tasks in building the simulation program was to check the experimental results against Sedgewick's formulas. This led to the discovery of an error in Sedgewick's analysis of quantity D, the number of insertions performed during the Insertion Sort phase. For median-of-3 Quicksort, D has the recursive form

$$D_N = 2\sum_{s=1}^{N} p_s D_{s-1} \qquad N > M,$$

where $p_s = \dfrac{(N-s)(s-1)}{\binom{N}{3}}$

$$D_N = N - H_N, \qquad N \le M$$

Sedgewick, in his thesis, gave the solution to this recurrence as

$$D_N = (N+1) - \frac{4(N+1)}{7(M+2)}(3H_{M+1} - 1).$$

This solution is incorrect: the last term should be $(3H_{M+1}+1)$. Although this is clearly a minor algebraic error, the resulting computation of D is off by a factor of N and gives an erroneous calculation of the optimum value of M. A later paper by Sedgewick [7] gives the correct solution but fails to carry the correction through to the calculation of M.

$$D_N = (N+1) - \frac{4(N+1)}{7(M+2)}(3H_{M+1} + 1).$$

Sedgewick's efficient MIX implementation of Quicksort has average running time

$$(53/2)A + 11B + 4C + 3D + 8E + 9S + 7N.$$

Solving with the correct formula for D and regrouping terms gives

$$-\frac{47}{2} + \frac{372}{35}(N+1)H_{N+1}$$
$$+(N+1)\times\left[\frac{529}{49} + \frac{18}{M+2} - \frac{372}{35}H_{M+2} - \frac{36}{7(M+2)}H_{M+1} + \frac{48}{35}M + \frac{27(5M+3)}{7(2M+3)(2M+1)}\right],$$

which differs from Sedgewick's formula in the first and fourth terms (which he has as $-111/2$ and $-450(N+1)/(7(M+2))$, respectively). Exhibit 5-1 shows $F(M)$, the function defined by the terms inside the large brackets, as M varies. The graph shows that the average running time of Sedgewick's MIX implementation is minimized when $M = 7$, not 9 as reported.

M	F(M)
1	-1.832653
2	-5.028571
3	-6.799184
4	-7.819406
5	-8.378633
6	-8.626538
7	-8.650244
8	-8.504810
9	-8.227213
10	-7.843478
11	-7.372606
12	-6.828899
13	-6.223390
14	-5.564770
15	-4.860008

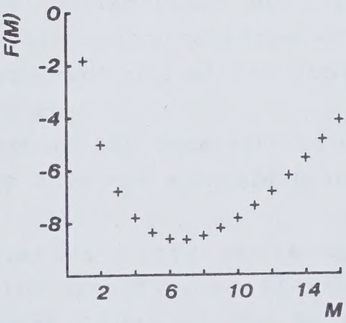


Exhibit 5-1: Minimizing F(M)

6. Conclusions

This paper presents a version of Quicksort that allows sample size for median selection to vary with sublist size, with evidence that it outperforms fixed-T schemes. The tradeoffs between partition comparisons and median-selection comparisons are also examined for fixed-T strategies. A number of open problems remain. It appears to be difficult to find a closed form for C, and even harder to derive the $t(n)$ that minimizes the number of comparisons. Although comparison cost was the primary measure in the simulations, the behavior of other measures is also of interest. An obvious next step is to study the actual running time of a "square root" strategy Quicksort. It is likely that the taking of square roots would dominates the computation time, in which case either a table-lookup scheme or a fast integer approximation of the square root function could be used.

References

1. J. L. Bentley. "Progamming Pearls: Selection". *Communications of the ACM 28*, 11 (November 1985).

2. C. A. R. Hoare. "Partition (Algorithm 63), Quicksort (Algorithm 64), and Find (Algorithm 65)". *Communications of the ACM 4*, 7 (July 1961), 321-322.

3. C. A. R. Hoare. "Quicksort". *Computer Journal 5*, 4 (April 1962), 10-15.

4. D. E. Knuth. *The Art of Computer Programming: Volume 2 Seminumerical Algorithms.* Addison-Wesley Publishing Company, Reading, MA, 1973.

5. D. E. Knuth. *The Art of Computer Programming: Volume 3 Sorting and Searching.* Addison-Wesley Publishing Company, Reading, MA, 1973.

6. Knuth. Solution to Exercise 5.2.2-32. In *The Art of Computer Programming: Volume 3 Sorting and Searching*, Addison-Wesley, Reading, Massachusetts, 1973, pp. 610-611.

7. R. Sedgewick. "Analysis of Quicksort programs". *Acta Informatica 7*, 4 (1977), 327-355.

8. R. Sedgewick. *Quicksort*. Ph.D. Th., Stanford, 1975.

9. R. C. Singleton. "An efficient algorithm for sorting with minimal storage (Algorithm 347)". *Communications of the ACM 12*, 3 (March 1969), 185-186.

TRANSITIVE CLOSURE FOR DYNAMIC GRAPHS*

G. F. ITALIANO
Dipartimento di Informatica, Universita' di Roma,
via Eudossiana 18, 00184 Roma, Italy.

ABSTRACT We consider a data structure to maintain a digraph with n vertices under an arbitrary sequence of two kinds of operations: an add operation that inserts an edge in the graph, and a searchpath operation that checks the presence of a path between a pair of vertices. Our data structure requires O(nlogn) time per operation when the time is amortized over a sequence of operations.

1. INTRODUCTION

To easily extract information about paths while updating a graph is a fundamental problem in Transportation, Transmission and Communication Networks, where the reachability relationship plays an important role (5), but it seems also to have significant applications in the representation of transitive implications in information retrieval systems.

We present a new data structure that starting with an empty graph (a graph without edges) of n vertices performs an arbitrary sequence of m add, searchpath instructions in O(mnlogn) worst-case time, thus requiring O(nlogn) time per operation when the time is amortized over a sequence of operations. This bound favourably compares with other data structures, for which at least one of the above operations requires $O(n^2)$ worst-case time.

A high-level description of the data structure now follows. The reader is required to know the standard graph terminology as contained in (6).

First of all we organize every cycle by means of a multilayer representation (see fig. 1). In what follows we shall refer to the upper layer as UL, to the vertices

---

* This work was supported by MPI within the National Project on Theory of Algorithms and by Selenia S.p.A.

corresponding to a cycle in the original digraph as C-vertices and to both vertices and C-vertices as nodes. In order to properly reconstruct a path within a multilayer representation, every edge leaving or entering a C-vertex is labelled with the name of the vertices it actually joins.

The second idea is to represent both transitive closure and transitive reduction of the digraph pointed out by the user during the add operations. In particular, we partition the edges of our graphs into two classes: solid and broken. Solid edges belong to the transitive reduction, while we represent redundant edges of the transitive closure as broken. Beyond, we shall refer to the edges incident on a node x in the transitive closure as the closure of x.

Thirdly, we solve our problem with a lazy approach. In fact, during the insertion of new edges, we shall recover the transitive closure (that (7) describes as a tedious work), trying to spread it over time. A simple way to accomplish this task is shown in fig. 2. When an edge (i,j) must be inserted, rather than connecting all the ancestors of i with all the descendants of j, we introduce a new-type vertex w, called lazy-node, and we link them through it. In the lazy-node w corresponding to the edge e, the following information is stored (see fig. 3): head(w)=h(e), tail(w)=t(e), and degree(w) being the number of links incident on w.

Note that we let links be from nodes to lazy-nodes and viceversa, but we shall not join together lazy-nodes. In the remainder of this paper we shall refer to these links as lazy-arcs, while L will denote the set of lazy-nodes introduced.

The presence of a lazy-arc (x,w) shows that we have not recovered the transitive closure from x to all the descendants of tail(w). Clearly, the introduction of lazy-arcs causes a computational degrading in the data structure: we shall try to recover it while processing new searchpath instructions.

In fig. 4 it is represented the upper layer of one of the graphs we have so far defined. In order to efficiently perform path searching in graphs like this, we represent them by means of adjacency dictionaries, i.e. data structures that allow us to access, to insert and to delete an item from a set in logarithmic time (3),(4). On the other hand, we require the possibility of searching in both directions and so a threaded representation is used. Thus we need six dictionaries for each vertex (see fig.5).In the figure,the following dictionaries

$S\_SUCC(x)=\{y \in V \ / \ \exists$ a solid edge $(x,y)\}$

$B\_SUCC(x)=\{y \in V \ / \ \exists$ a broken edge $(x,y)\}$

$LAZYSUCC(x)=\{w \in L \ / \ \exists$ a lazy-arc $(x,w)\}$

contain the nodes and lazy-nodes adjacent from vertex x, while the remaining dictionaries store the threads (i.e. the nodes and lazy-nodes adjacent to x) in a similar way. Fig. 5 also applies for C-vertices if we add a link for joining a C-vertex to the appropriate lower-layer cycle representation (see fig. 1). In such a framework, the presence of a path between vertices i and j can be checked as follows:

```
procedure searchpath(i,j,T);
begin   path(i,j,T,UL); expand(T)   end searchpath;
```

where the procedure path(i,j,T,B) searches a path T from i to j in the layer B (UL stands for the upper layer), while the procedure expand eliminates from the path T all the C-vertices it contains by replacing them with the properly reconstructed paths inside the lower layers. For istance, fig. 6 shows how the searchpath algorithm works while checking a path from 8 to 11 in the multilayer representation of fig. 1. Clearly, we preliminary need to know that inside UL vertex 8 is represented by C-vertex A and 11 by B, while C stands for 5 inside B_1. This information can be organized defining leader(u,B_i) as the node standing for the vertex u inside the layer B_i and associating to each layer a sorted leader table with the key entries set to the vertices of the subgraph represented by B_i (see fig. 7).

To complete our high-level description we show how to deal with the insertion of edges in our graphs. According to the multilayer structure, if an edge (i,j) has to be inserted we effectively try to insert the properly labelled edge (u,v)=(leader(i,UL),leader(j,UL)). However, we must insert it only if there was no previous path from u to v since if such a path exists there is also a (solid or broken) edge (u,v).

If there was no path from u to v the insertion of this edge could give rise to a new cycle which can be detected as follows. Since a new cycle can be introduced only if there was at least a reversal path from v to u, let X contain all the UL nodes that are both ancestors of u and descendants of v (for each $x \in X$, $v \xrightarrow{*} x \xrightarrow{*} u$ and so x is in such a reversal path). More formally, X contains all the UL nodes that after inserting (i,j) will be in a new cycle. Thus, if X is an empty set there will be no cycle after the insertion of (i,j) and so the edge (u,v) can effectively be inserted. Otherwise we have to shrink X, replacing it with a single C-vertex.

Furthermore both transitive closure and reduction have to be maintained. Setting P (S) to the upper layer ancestors of i (descendants of j), this task can be accomplished by inserting a broken edge from P to S. In this way if there were already solid edges from P to S we break them, thus not allowing redundant edges to be solid. Since the upper layer is acyclic the transitive reduction is recovered (1), while (7) assures that also the transitive closure has been properly achieved. Unfortunately, this operation can involve too many updates as to require $O(n^2)$ worst-case time, depending on the size of P and S. We decide to do it only when these sizes are reasonably small ($|P| \cdot |S| \leq n$). In the opposite case we are pleased with maintaining a pseudoclosure (e.g. the transitive closure is recovered in a lazy way as shown in fig. 2). In conclusion, an edge can be inserted as follows:

```
procedure add(i,j);
begin path(i,j,T,UL);
      if $T=\emptyset$ then
         begin P:=anc(leader(i,UL));
               S:=desc(leader(j,UL));
               X:=$P \cap S$;
               if $X \neq \emptyset$
                  then cycle(X,(i,j),P,S)
                  else insertion(i,j,P,S);
               if $|P| \cdot |S| \leq n$
                  then closure(P,S)
                  else pseudoclosure(i,j,P,S)
         end
end add;
```

where anc(x)=B_PRED(x)$\cup$S_PRED(x), desc(x)=B_SUCC(x)$\cup$S_SUCC(x).

In this shortened version of the paper we omit many details of our ideas. However, a detailed timing analysis can be found in (7).

2. REPRESENTATION OF CYCLES

As far as the cycle representation is concerned, an obvious solution would be to store a minimum equivalent graph of the digraph, a problem which is known to be NP-complete (8). The idea we use is quite different. Since a cycle was caused by the insertion of a new edge, say it (x,y), while maintaining all the edges that are internal to the cycle, we separately store (x,y) by letting a basic edge variable for each lower layer and setting it to (x,y). That is, we represent every cycle in an acyclic way. For example, if the edge (3,4) caused the cycle in fig. 7, we represent it as shown in fig. 8. Hence we can properly reconstruct a path from a vertex u to a vertex v within a layer B_i by examining the leader table in B_i and

- by searching a path from leader(u,B_i) to leader(v,B_i), or in case of failure
- if B_i is a lower layer and (x,y) is its basic edge, by searching a path from leader(u,B_i) to leader(x,B_i) and from leader(y,B_i) to leader(v,B_i).

Clearly, a new cycle X must be organized as follows:

1. By substituting X in UL by a unique C-vertex c.
2. By initializing a new layer for representing X in a complete way.

A detailed timing analysis (7) shows that this induces a total of O(mnlogn) worst-case time over a sequence of m add, searchpath operations.

3. INSERTING AN EDGE

The procedure insertion shows how the edge (i,j) can be added in our multilayer structure:

INPUT : (i,j), the edge which has to be inserted; P, the UL ancestors of i; S, the UL descendants of j.

```
procedure insertion(i,j,P,S);
begin x:=leader(i,UL); y:=leader(j,UL);
      insert a solid edge (x,y); P:=P-{x}; S:=S-{y};
      if ((x≠i)or(y≠j)) then label (x,y) with (i,j);
      for each p on P do insert the broken edge (p,y);
      for each s on S do insert the broken edge (x,s)
end insertion;
```

OUTPUT : P and S, the UL nodes between which the transitive closure and transitive reduction requirements have to be recovered.

As we have previously specified, both transitive closure and reduction can be simply recovered as follows (see fig. 9):

```
procedure closure(P,S);
begin for each (u,v) on PxS do insert a broken edge (u,v)
end closure;
```

but we can also accomplish this task lazily (see fig. 10):

```
   procedure pseudoclosure(i,j,P,S);
1. begin create a new lazy-node w; head(w):=leader(i,UL);
2.       tail(w):=leader(j,UL); degree(w):=|P|+|S|;
3.       P:=P-anc(leader(j,UL); S:=S-desc(leader(i,UL);
4.       for each x on P do insert a lazy-arc (x,w);
5.       for each y on S do insert a lazy-arc (w,y)
   end pseudoclosure;
```

depending on the sizes of P and S. In this lazy version a new lazy-node w is created and associated with (i,j) (i.e. we set head(w) to i and tail(w) to j). Since the presence of a lazy-arc (x,w) shows that the dictionaries in x were not updated during the insertion of the edge (head(w),tail(w)), we shall not insert such a lazy-arc if there was a previous path from x to tail(w) (in fact in such a case the insertion of (head(w),tail(w)) does not add further information into the dictionaries in x). In a similar way if y knows head(w) as its own ancestor, we shall not insert a lazy-arc (w,y). This goal can be achieved as lines 3-5 show.

4. SEARCHING A PATH

In this section we shall see how to recover the closure of a node. This result will be used to find paths very quickly.

Let at least one lazy-arc be incident on a node x, say it (x,w) (this implies that x does not know anything about all the descendants of tail(w)). The information we need to recover the work pointed out by (x,w) can be found by means of a forward search from tail(w) with the following rules:

(r1) only solid edges are considered;

(r2) when we reach a node v which has already been visited, either its descendants are all known to x or there will be another lazy-arc leaving x and pointing to them. In any case we must stop. If v was solid-adjacent from x, we have to make it broken-adjacent since the redundant edge (x,v) can no longer be solid;

(r3) when an unvisited node u is reached, we simply insert a broken edge (x,u).

At the end of this search process the lazy-arc (x,w) can be deleted since all the work pointed out by it has already been done. A lazy-arc (w,x) can be recovered in a similar way.

This recover algorithm can be used to check the presence of a path within any layer, as the procedure path shows. If a path from i to j in the layer B has to be checked, first of all the closure of x=leader(i,B) and y=leader(j,B) is recovered (line 2). Then all the nodes of the layer B in a path from x to y can be found by defining A as desc(x)∩anc(y) (note that since the closure of x and y has just been recovered, desc(x) and anc(y) actually contain all the descendants of x and the ancestors of y in the layer B). If A is an empty set y is not reachable from x, otherwise a path from x to y can be found by means of an informed search (lines 4-13). In such a search if a node v which has already been visited is reached, there is a solid edge entering v (say it (u,v)) which is redundant, i.e. $u \xrightarrow{+} v$ (note that it may happen as sometimes the transitive reduction is recovered in a lazy fashion, as line 11 of the procedure add shows). If for each examined node v a reversal pointer p(v) is maintained, in this case the solid edge (p(v),v) must be broken (lines 9-11).

```
procedure path(i,j,T,B);
begin x:=leader(i,B); y:=leader(j,B);
      recover(x); recover(y);
      A:=desc(x)∩anc(y); T:=∅; u:=x;
      if A≠∅ then
      begin while u≠y do
            begin SONS:=S_SUCC(u);
                  repeat
                     select any v on S;SONS:=SONS-{v};
                     if v has already been examined
                         then break the edge (p(v),v);
                     p(v):=u
                  until v∈A;
                  T:=T∘(u,v); u:=v
            end
      end
end path;
```

Once a path has been found, we can expand the C-vertices it contains as the procedure expand shows:

```
procedure expand(T);
begin while there is a C-vertex z in T do
      beginlet (u,v), (x,y) be the labels of the edges in T
            entering and leaving z, Z be the layer pointed by
            z and (h,k) the basic edge in Z;
            path(v,x,P1,Z);
            if P1=∅ then begin path(v,h,P2,Z);path(k,x,P3,Z);
                                 P1:=P2∘(h,k)∘P3
                          end;
            replace z in T by P1
      end
end expand;
```

A detailed analysis spelled out in (7) shows that the procedures add and searchpath perform an arbitrary sequence of m operations in O(mnlogn) worst-case time.

REFERENCES

(1) Aho A.V., Garey M.R. and Ullman J.D., "The transitive reduction of a directed graph", SIAM J. Comp., 1 (1972), 131-137.

(2) Aho A.V., Hopcroft J.E. and Ullman J.D., The design and analysis of computer algorithms, Addison-Wesley, 1974.

(3) Adelson-Velskii G.M. and Landis Y.M., "An algorithm for the organization of information", Soviet Math. Dokl., 3 (1962), 1259-1262.

(4) Bayer R. and McCreight E., "Organization and maintenance of large ordered indices", Acta Informatica, 1 (1972), 173-179.

(5) Frank H. and Frish I.T., Communication, transmission and transportation networks, Addison-Wesley, 1971.

(6) Harary F., Graph theory, Addison-Wesley, 1972.

(7) Italiano G.F., "Il problema della chiusura transitiva di grafi: algoritmi e strutture di dati efficienti", Thesis Dip. Inform., University of Rome, Rome, Italy, 1986.

(8) Moyles D.M. and Thompson G.L., "Finding a minimum equivalent graph of a digraph", J. ACM, 16 (1969), 455-460.

FIGURES

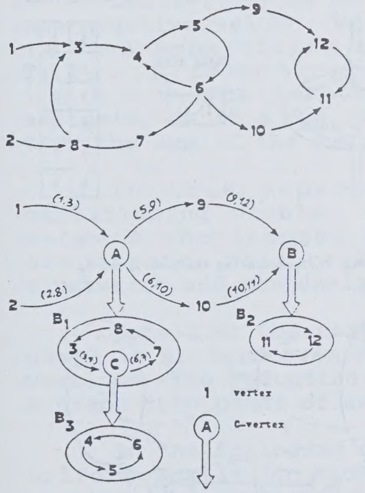


Fig. 1. A multilayer representation of a digraph

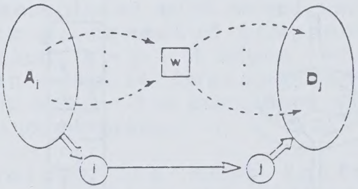


Fig. 2. Recovering the transitive closure in a lazy way

head(w)	degree(w)	tail(w)

Fig. 3. Storing a lazy-node w corresponding to the edge (head(w),tail(w)); degree(w) is the number of lazy-arcs incident on w

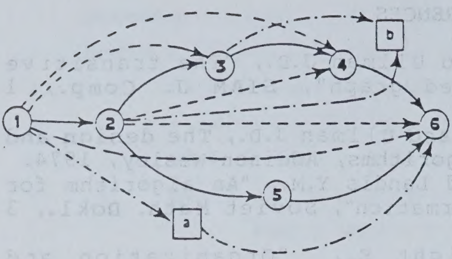

b
3
4
1
2
6
5
a
node
lazy-node
solid edge
broken edge
lazy-arc

Fig. 4. Representing the upper layer

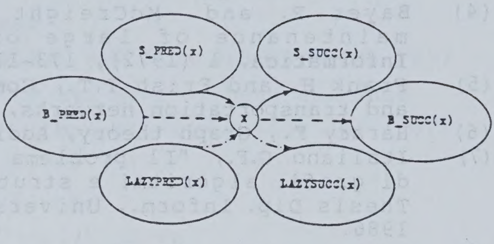

S_PRED(x)
S_SUCC(x)
B_PRED(x)
x
B_SUCC(x)
LAZYPRED(x)
LAZYSUCC(x)

Fig. 5. Representing a vertex x

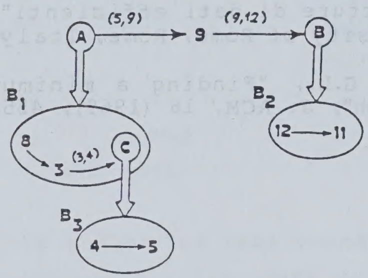
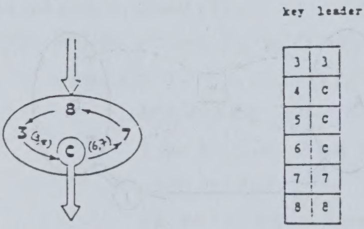
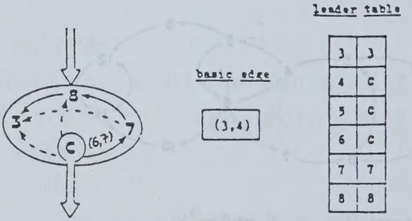

(5,9)
(9,12)
A
9
B
B1
B2
8
C
3
(3,4)
12
11
B3
4
5

Fig. 6. Searching a path from 8 to 11 in the graph of fig. 1



key	leader
3	3
4	C
5	C
6	C
7	7
8	8

Fig. 7. The leader table of the layer B_1 in fig. 1


8
3
7
C
(6,7)
basic edge
(3,4)

leader table

3	3
4	C
5	C
6	C
7	7
8	8

Fig. 8. Representing the lower layer B_1

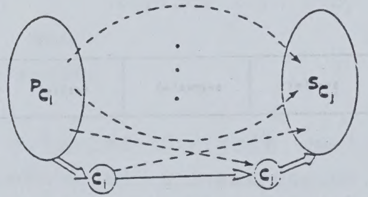

P_{C_i}
S_{C_j}
C_i
C_j

Fig. 9. Recovering the transitive closure

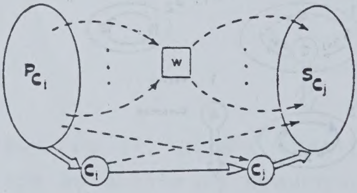

P_{C_i}
w
S_{C_j}
C_i
C_j

Fig. 10. Recovering the transitive closure in a lazy way

HOW HARD IS IT TO PARTITION A PATH

MARIO LUCERTINI[1]
YEHOSHUA PERL[2]*
BRUNO SIMEONE[3]

1. Faculty of Engineering,
Second University of Rome.

2. Computer and Information Science Department
New Jersey Institute of Technology,
Newark, New Jersey 07102.

3. RUTCOR, Rutgers University,
New Brunswick, New Jersey 08903 and
Department of Statistics, University of Rome.

Abstract:

Consider the most uniform p-partitioning of a node where we minimize the difference between the weights of the heaviest and lightest components. We first consider an easier problem,the (A,B)-p-partitioning where the weight of each component is between A and B. A linear algorithm based on a sophisticated preprocessing procedure is presented. Using n^2 applications of the (A,B)-partitioning yield an $O(n^3)$ algorithm for the most uniform partitioning of a path.

1. INTRODUCTION:

Let T=(V,E) be an (unrooted) tree with n vertices. A non-negative weight W(v) is associated with every vertex $v \in V$. A p-partition of T into p connected components $T_1, T_2, .., T_p$ is obtained by deleting $k = p - 1$ edges of T, $1 \leqslant k < n$. The deletion of an edge is implemented by assigning to it a cut. The weight $W(T_i)$ of a component T_i is then the sum of the weights of its vertices.

In this paper we consider the most uniform partitioning problem where we minimize the difference between the largest and the smallest weights of the components. This problem has applications in paging, clustering and the design of communication networks.

Note that the complexity of the problem for a general graph is NP-hard even for the case of unit weights for the vertices. The reduction is from the problem of partitioning a graph into paths of length 2. [5]

In the following discussion we shall consider the most uniform partitioning problem in the context of the other partitioning problems for trees.

---

* Supported in part by the State of New Jersey SBR Grants.

In previous papers we consider different optimization criteria for tree partitioning.In [7] a shifting algorithm is presented for the max-min partitioning problem where the weight of the minimum component is maximized. In [2] another shifting algoritm is given for the min-max partitioning problem where the weight of the maximum component is minimized. A more efficient implementation of this algorithm appears in [9]. Generalization of those two algorithms for general weighting functions appear in [3].

Actually the most desired partitioning criterion for many applications is the uniform partitioning where each of the p-components has the same weight. But such a partition does not always exist. One actually realizes that the max-min and the min-max partitioning are actually approximations for the uniform partition. To see this note that in case a uniform partition exists, it is obtainable both as a max-min partition and as a min-max partition. In [4] we consider another approximation for the uniform partition, the equi-partition of a tree where we minimize the sum of the differences between the weights of the components and the weight of the average component. However it is shown that the equi-partitioning problem for a general tree is NP-hard. Algorithms are presented for special families of trees.

A better approximation for the uniform partition is the most uniform partition where we minimize the difference between the largest and smallest weights of the components. If this difference is zero then the uniform partition is obtained.

The problem of finding the most uniform partition is difficult since in order to minimize the difference of the weights of the heaviest and lightest components one needs to take simultaneously into account both the weight of the heaviest component and the weight of the lightest component. The experience is that when one tries to optimize on two parameters at the same time the problem gets more difficult. For example in [1] we consider two weighting functions at the same time. It is shown that minimizing the weight while having a given constraint on the size of the components makes the problem NP-hard. On the other hand for the problem of minimizing the weight while having a given constraint on the height of the components a very complicated polynomial shifting algorithm is presented. In [8] we consider tree partitioning into minimum number of components under given constraints for both the size of each component and the capacity of the connections of each component, where each vertex has a given size and each edge has a given capacity. This problem is shown to be NP-hard and a pseudo polynomial algorithm is given.

In order to deal with an objective function involving both the largest and the smallest weights of the components in a partition we define related problem - the (A,B) - partitioning problem where the weight of each components must be between the values A and B, A < B. This problem is also interesting by itself and has applications in paging and clustering. The related problem of partitioning a tree into the minimum number of components such that the weight of each components is at most B was solved in linear time in [6]. A linear algorithm for another related problem of partitioning a tree into the maximum number of components such that the weight of each component is at least A is given in [7]. Howevere the problem when the components have to satisfy both bounds A and B is much more complicated. Thus we decided as a first step to analyze the problem for the special case of a path. To our surprise, even this restricted problem is still quite complicated and interesting by itself.

In Section 2 we demonstrate the complications arriving in the problem of (A,B)-partitioning of a path by discussing some straightforward approaches. A path preprocessing technique is presented in Section 3. In Section 4 linear algorithms are presented for three related problems.

1. (A,B) partitioning into the minimum number of components denoted by minimum (A,B)-partitioning.
2. (A,B) partitioning into the maximum number of components denoted by maximum (A,B)-partitioning.
3. (A,B) partitioning into p components denoted by (A,B)-p-partitioning.

The algorithm for the last problem uses the solutions for the two previous problems.

Finally we use the above linear algorithm for (A,B)-partitioning to obtain a fast polynomial algorithm for the most uniform partitioning of a path. Actually, the later problem turns out to have important applications in image processing. The problem is how to group many tones of grey into bands formed by contiguous tones of grey. It has been found empirically that the best optical contrast in the output image occurs when the frequencies of the bands are most uniform.

2. STRAIGHTFORWARD APPROACHES:

One might think that when we restrict ourselves to (A,B)-partitioning of a path,the problem is simple. Actually it is not. However, we know that the problem is solvable in polynomial time since there exists an $O(n^2k)$ "dynamic programming" algorithm for it.

However the complexity of the dynamic programming procedure is too high. This is especially true since our main purpose is an efficient algorithm for most uniform partitioning of a path. As we shall see in section 4 many applications of the (A,B)-partitioning algorithm are required for finding a most uniform partition. Thus we look for a more efficient algorithm for the (A,B)-partitioning of a path.

For this purpose we shall consider a greedy approach. We shall start by discussing a straightforward greedy technique. Let us first consider the minimum (A,B)-partitioning problem which is simpler for a greedy approach. It is easy to obtain a minimum partition such that the weight of every component is at most B. To obtain such a partition we perform a linear scanning of the path, accumulating the weight of a component and assigning a cut to the edge leading to the vertex whose weight added to the component weight would result a weight exceeding B. However we have to be careful bout satisfying the other requirement that the weight of each component is at least A. We shall demonstrate this approch by applying it to an example. This will also serve to demonstrate the complication of the problem and develop some intuitive insight into its structure. This insight will help us in understnding our final algorithm in Section 4.

In figure 1(a) we see that after assigning 3 cuts we are stuck since we can not assign the fourth cut and still have an (9,13)-partition since either the forth component is too light or too heavy. Thus we must move some previously assigned cuts. When we assigned the fourth cut, the fifth component becomes too light and we start to shift the previous cuts to the left to satisfy the requirements. We adopt here a stingy approach shifting each cut as little as possible. So we shift only the cut C_3 and then we are able to continue to assign C_5 and C_6, obtaining the partition in Figure 1(b). But now we are stuck again since the seventh component is either too light or too heavy. So we apply again the shifting procedure to rearrange the previous cuts. Actually only C_6 is shifted one position to the left. Finally we can assign C_7 to obtain the partition in Figure 1(c).

Now we are almost done but there is another problem. The last component is too heavy but can not be partitioned into two components. Thus we have to apply again the shifting procedure to rearrange the previous cuts. In this case all seven cuts are shifted and then the eight cut can be assigned to obtain the (9,13)-9- partition in Figure 1(d). Note that this partition is, by coincidence, a maximum (9,13)-partition as well as a minimum (9,13)-partition. So it is the only feasible (9,13)-partition for this path.

In this example the greedy approach which we applied gave the desired solution. However the example indicates a problem affecting the complexity of the algorithm: each time we add a cut we might have to shift all previous cuts. This implies that the complexity of the procedure would not be low enough. In addition to these there are more difficulties which are not demonstrated in this example.

Thus, a deeper analysis of the problem is necessary for the design of the algorithm. Such an analysis is presented in the next section.

3. THE PREPROCESSING PROCEDURE:

Let us review the procedure described in Section 2 Suppose the scanning gets stuck after the cut C_i is assigned to the position j between the j-th and the j+1-th vertices. The reason why we can not assign the next cut C_{i+1} is because up to some position, say m, the accumulated weight of the current component is lighter than A and if we add the next weight W(m) the component gets heavier than B. Thus we can conclude that no (A,B)- partition may have a cut at the position j; that is, position j is forbidden. Note that this conclusion is independent of the assignment of other cuts to the left or to the right of the j-th position. Note that in the example of Figure 1 every time the scanning got stuck there was an attempt to assign a cut to a forbidden position. We can prevent the scanning from getting stuck by not assigning a cut to a forbidden position. Or alternatively by combining the two vertices to the left and to the right of the forbidden position into one vertex with a weight which is the sum of the weights of the two vertices. We call this operation compression. We shall show later that if a weight larger than B is obtained this way then there is no feasible solution for our problem.

Thus we should apply a preprocessing of the path to check whether each position is forbidden or not and perform the appropriate compressions for the forbidden positions.

After applying the compressions there may be new forbidden positions in the compressed path due to the first compressions which cause to add the larger compressed weight at one time rather than being able to stop after adding some of the single original vertices. After we deal with those new forbidden positions by further compressions we may obtain new forbidden positions and so on. Thus the complexity of the preprocessing will be too high.

But we can perform the preprocessing in a more efficient way. Actually a position, say m, which was not forbidden became forbidden due to a compression done to the right of the position. This compression happened due to a forbidden position to the right of the position m. If we process the path from right to left rather than from left to right we will identify a position which becomes forbidden due to another forbidden position, immediately in the first phase rather than in later phases. In Figure 2 we demonstrate the right to left preprocessing of the same example of Figure 1. Note that we still consider the weights to the right of a position as before, only the order of processing the positions is changed.

In the preprocessing procedure all forbidden positions are found and all required compressions are done in one phase. As we show later, after this one phase there are no more forbidden positions. In this example the preprocessing already compressed the vertices to vertices of weights between A and B and thus obtaining an (A,B)-partition.

Lemma 1: For any (A,B)-partition of a path there exists a corresponding (A,B)-partition of the path obtained by a compression operation of the preprocessing procedure, where the correspondence is equality up to a compression of two vertices.

Lemma 2: If a weight larger than B is obtained during the preprocessing of a path, then there is no (A,B)-partition of the original path.

Lemma 3: There are no forbidden positions in the path obtained by the preprocessing.

Note that our treatment is assuming all the time that the component considered is to the right of the position checked and later the scanning is from left to right. Of course it is possible to assume the reverse order from right to left and then have different forbidden positions and compressions. However as our proofs show we do not need to work in both directions; so we have only a left to right treatment.

The complexity of the preprocessing is not clear since we cannot know how many weights we have to add till the sum exceeds A. If all weights are integers then we add each time no more weights than A. This implies the trivial bound O(n A) for the complexity of the preprocessing.

However, it is possible to reorganize the preprocessing such that the complexity is linear. The idea is to use the weight accumulated for checking the position m in order to compute the weight necessary for checking the position m-1 (Note that the order of the preprocessing is still from right to left). To compute the accumulating weight for the position m-1 we add to the weight accumulated for the position m the weight W(m) and on the other hand we subtract the last weights from the accumulated weight until it gets into the range between A and B. In case the accumulated weight does not eventually get into the range between A and B then the position m-1 is forbidden and a compression of v(m-1) and v(m) is performed.

We describe now the preprocessing procedure. The index POS points to the vertex to the right of the position being currently checked. The weights of the vertices in the component to the right of the position POS are inserted into the queue C with sum of the weights WC. Compressions are done while entering the queue where a new weight is added to the last inserted weight. The weights are checked if they are larger than B (i.e. no (A,B)-partition) while being deleted from the queue. The weights are then transfered to a second array NW filling the locations I+1,..N. Note that due to the compressions there are less weights in NW than in W.

Note that even with this implementation we cannot bound by a constant number the number of operations done to check a given position, since the number of weights deleted from the queue during this process is not fixed. However, we can use the concept of amortizing complexity analysis formalized by [10].During the operation of the preprocessing procedure each weight was inserted and deleted from the queue exactly once. Thus the procedure is linear.

4. PARTITIONING OF THE PREPROCESSED PATH:

In Section 2 we tried the straightforward approach of partitioning by scanning the path while accumulating the weights. The problem was that this process got stuck due to forbidden positions. Now that all forbidden position are eliminated by the preprocessing we shall show that the straightforward approach cannot fail.

Define an (A,B)-scanning as a scanning of the path for which each time a cut is assigned the weight of the current component is between A and B. Note that an (A,B)-scanning still does not determine the position of the cuts since there is still freedom to assign a cut to probably several positions from the time the accumulated weight reaches A until it exceeds B. For example a greedy (A,B)-scanning assigns a cut just before the accumulated weight exceeds B. On the other hand a stingy (A,B) scanning assigns a cut as soon as the accumulated weight reaches A.

Theorem 4: No (A,B)-scanning of a preprocessed path ever gets stuck.

Theorem 5: The greedy (A,B)-scanning of a preprocessed path yields a minimum (A,B)-partition.

Theorem 6: The stingy (A,B)-scanning of a preprocessed path yields a maximum (A,B)-partition.

The minimum and maximum (A,B)-partition of a path are demonstrated in Figure 3(a) and 3(b), respectively, where the minimum partition has 6 components and the maximum partition has 9 components.

There is still a question how do we find an (A,B)-partition into k-components where k is any number between the minimum and the maximum number of the components. The problem is that in these cases we do not have the greedy or stingy approach to guide the scanning. However we can use the minimum and maximum partitions to obtain an (A,B)-k-partition for any k between the minimum and the maximum.

Suppose the minimum number and the maximum number of components of an (A,B)-partition are m and n respectively. We look for an (A,B)-p-partition for any number p such that $m < p < n$. Let $d = p - m$. Let min-cut denote a cut of a minimum partition and let max-cut denote a cut of a maximum partition. Note that there are n-1 max-cuts and m-1 min-cuts to the left of this position except the last such max-number of min-cuts and the number of max-cuts upto each position of the path. When we get to a position of a min-cut such that the difference between the number of max-cuts and min-cuts is d then the partition obtained by using the max-cuts to the left of this position except the last such max-cut(which may be at the same position of the above min-cut) and the min-cuts to the right of this position starting at this position itself.

Theorem 7: The partition obtained from a minimum (A,B)-partition and a maximum (A,B)-partition as described above is an (A,B)-k-partition.

In Figure 3(c) and 3(d) a (3,5)-7-partition and a (3,5)-8-partition are given. They are obtained from the minimum (3,5)-partition and the maximum (3,5)-partition given in Figure 3(a) and 3(b), respectively. Note that the path in Figure 3 is such that the preprocessing procedure does not perform any compression.

Both the greedy and the stingy scanning procedures to obtain the minimum and maximum (A,B)-partitions, respectively, are linear. Furthermore the complexity of obtaining an (A,B)-p-partition from the minimum and maximum (A,B)-partitions is also linear. Hence, the complexity of finding an (A,B)-p-partition of a path is linear.

Finally one can solve the most uniform partition (p-partition) by scanning a boolean table showing the (A,B)-partitionability (p-partitionability) for any possible pair (A,B) and looking for an entry minimizing the difference B - A. Note that since the weight of each component must in any case belong to the set $\{ w(i) + w(i+1) + \ldots + w(j) : 1 < i < j < n \}$, one is interested only in $O(n^2)$ possible values for A and B and hence in at most $O(n^4)$ entries in the table. Furhtermore, if an (A,B)-partition exists and $0 < A' < A < B < B'$, then an (A',B')-partition also exists. Due to this monotonicity property, it is possible to limit the scanning only to $O(n^2)$ entries. Thus the complexity of finding a most uniform partition (p-partition) of a path is $O(n^3)$.

```
(a) 11  2 |4 4  3 2 |5  6  2  |4  3* 7 2 3  4 3 2 2 3 5 6 11
          C1        C2        C3

(b) 11  2 |4 4  3 2 |5  6 |2  4  3 |7 2 3 |4 3 2 2 |3 5*6 11
          C1        C2    C3       C4     C5       C6

(c) 11  2 |4 4  3 2 |5  6 |2  4 3 |7 2 3 |4 3 2 |2 3 5 |6*11
          C1        C2    C3      C4     C5     C6     C7

(d) 11 |2  4 4 |3 2  5 |6  2 4 |3 7 |2 3 4 |3 2 2 3 |5 6 |11
       C1      C2      C3      C4   C5     C6       C7   C8
```

Figure 1: Straight Forward (9,13)-scanning.

```
  11   2   4   4   3   2   5   6   2   4   3   7   2   3   4   3   2   2   3   5  *6  11
                                                                       2  *3  11  11
                                                                   2  *5  11  11
                                                               3  *7  11  11
                                                       3  *4  10  11  11
                                                   2  *7  10  11  11
                                           3  *7   9  10  11  11
                                   2  *4  10   9  10  11  11
                               6  *6  10   9  10  11  11
                       2  *5  12  10   9  10  11  11
                   3  *7  12  10   9  10  11  11
               4  *4  10  12  10   9  10  11  11
       2  *8  10  12  10   9  10  11  11
  11  10  10  12  10   9  10  11  11
```

Figure 2: Right to Left Preprocessing.

```
1  2  1 |2  1  2 |1  2  1 |2  1  2 |1  2  1 |2  3
```

(a) Minimum (3,5) 6-partition.

```
1  2 |1  2 |1  2 |1  2 |1  2 |1  2 |1  2 |1  2 |3
```

(b) Maximum (3,5) 9-partition.

```
1  2 |1  2 |1  2 |1  2  1 |2  1  2 |1  2  1 |2  3
```

(c) (3,5) 7-partition.

```
1  2 |1  2 |1  2 |1  2 |1  2 |1  2 |1  2  1 |2  3
```

(d) (3,5) 8-partition.

Figure 3: (3,5)-partitions of a path.

REFERENCES:

1. Agasi,E.,Becker,R.I.,and Perl,Y.,A shifting algorithm for min-max constraint partition on trees,in the proceedings of the 16th Annual Conference on Information Science and Systems, Princeton, 1982.

2. Becker,R.I., Perl,Y.and Schach,S.R.,A shifting algorithm for min-max tree-partitioning. J. ACM 29(1982), 58-67.

3. Becker,R.I.and Perl,Y. Shifting algorithm for tree partitioning with general weighting functions, J.Algorithms, 4, (1983) 101-120.

4. De Simone,C.,Lucertini,M.,Pallottino,S.and Simeone,B., Fair dissections of spiders worms and caterpillars, presented in EURO VII Seventh European Congress on Operation Research. Bolonga, Italy 1985.

5. Garey, M. R. and Johnson, D. S. Computers and Intractability, Freeman, San Francisco (1979).

6. Kundu, S. and Misra, J., A linear tree-partitioning algorithm. SIAM J. Computing 6, 1(1977), 131-134.

7. Perl,Y.and Schach,S.R.,Max-min tree-partitioning. J. ACM 28, 1(1981), 5-15.

8. Perl,Y. and Snir, M., Circuit partitioning with size and connection constraints,NETWORKS 13(1983) 365-375.

9. Perl,Y. and Vishkin,U., Efficient implementation of a shifting algorithm, Discrete Applied Math 12(1985), 71-80.

10. Tarjan,R.E., Amortized computational complexity, SIAM J. Alg. Disc. Meth. 6(1985) 306-318.

A COMPARISON OF LOW COMPLEXITY ALGORITHMS FOR FINDING MINIMUM CYCLE CUTSETS

HANOCH LEVY
AT&T Bell Laboratories[1]
Holmdel, NJ 07733

ABSTRACT

We consider the problem of finding a minimum size cutset in a directed graph $G=(V,E)$, i.e., a vertex set that cuts all cycles in G. Several low complexity algorithms for addressing this NP-complete problem have been proposed in recent years. In this paper we compare these algorithms by studying the relations between the graph classes for which they guarantee finding minimum cutsets. We construct a hierarchy of the graph classes which implies an ordering of the algorithms according to their strength. As a byproduct of the hierarchy we gain more knowledge of the properties of quasi-reducible graphs.

1. INTRODUCTION

We consider the problem of finding a minimum size *cutset* in a directed graph $G=(V,E)$, i.e., a vertex set that cuts all cycles in G. Several low complexity algorithms for addressing this NP-complete problem have been proposed in recent years. Shamir[1] proposed a depth-first-search algorithm that finds minimum cutsets for reducible flow graphs. When the algorithm is not successful in finding a minimum cutset it aborts. Rosen[2] modified this algorithm to prevent it from aborting in these cases; instead, the modified algorithm will return in those cases a cutset which is not necessarily a minimum cutset. The graphs for which Shamir-Rosen's algorithm guarantees finding a minimum cutset are called *quasi-reducible*[2]. Wang, Lloyd and Soffa[3] [4] [5] suggested an $O(|E|\cdot|V|^2)$ algorithm which finds minimum cutsets for graphs which are called *cyclically reducible*. Levy and Low [6] [7] suggested an algorithm which identifies cutsets by performing contraction operations on the graph. The time complexity of the contraction algorithm is $O(|E|\log|V|)$ and the graphs for which it finds minimum cutsets are called *completely contractible*.

In this paper we deal with a comparison of these algorithms. We concentrate on comparing the classes of graphs for which the algorithms guarantee finding minimum cutsets. Previous comparisons by Wang, Lloyd and Soffa[4] [5] showed the following properties: 1) The class of completely contractible graphs properly contains the class of cyclically reducible graphs. 2) The classes of quasi-reducible graphs and cyclically reducible graphs are not related to each other, i.e., neither of the classes contains the other. The major problem left open in this context then is the comparison between the classes of quasi-reducible graphs and completely contractible graphs. The main difficulties in addressing this problem are the large difference (in nature of operation) between the contraction algorithm to Shamir-Rosen's algorithm and the "inconsistent behavior[3]" of Shamir-Rosen's algorithm.

---

[1] This work was done while the author was with IBM, Los Angeles Scientific Center.

[2] A finer definition of quasi-reducibility is given in the sequel.

[3] Inconsistent in the sense that the algorithm may or may not find a minimum cutset for a certain graph, depending on the particular search order chosen by the algorithm. See Rosen[2] for a discussion of this issue.

The main result derived in this paper is that the class of backward contractible graphs, which is a subset of the class of completely contractible graphs, properly contains the class of quasi-reducible graphs. This result allows us to fully compare the power of the three algorithms. First, it is directly implied that Levy-Low's algorithm is stronger than Shamir-Rosen's algorithm since the class of completely contractible graphs properly contains the class of quasi-reducible graphs. Second, we are able to compare Shamir-Rosen's algorithm to Wang-Lloyd-Soffa's algorithm by observing that the class of cyclically reducible graphs contains a class which is a "mirror image" of the class of backward contractible graphs and, thus, is larger than the class of quasi-reducible graphs. Third, from previous studies[5] it is implied that Levy-Low's algorithm is stronger than Wang-Lloyd-Soffa's algorithm.

As a by-product of this comparison we obtain that the class of quasi-reducible graphs is "squeezed" in between two classes of graphs (backward contractible and reducible) which are both defined in terms of contraction operations and which are very similar. This result illuminates the nature of quasi-reducible graphs which are otherwise defined in terms of an (inconsistently behaving) depth first search algorithm.

The structure of this paper is as follows: In Section 3 we describe the three algorithms, the contraction operations, and the classes of graphs relevant to the discussion. In Section 4 the main result of this paper, namely, that the class of backward contractible graphs contains the class of quasi-reducible graphs, is derived. In Section 5 we compare the classes of graphs and the three algorithms.

2. PRELIMINARIES

We consider a directed graph $G=(V,E)$ where V is a set of vertices and $E \subseteq V \times V$ is the set of directed edges. An edge e from v to u is denoted by $v \to u$, or by $v \xrightarrow{e} u$. In this case, we say that e *emanates* v and *enters* u. e is also said to be *incident* both to v and u. The *number* of edges emanating v is denoted by outdegree(v) and the number of edges entering v is denoted by indegree(v). A *path* from v to u is a sequence of zero or more edges in G of the form $v \to v_1 \to v_2 \ldots \to v_k \to u$ and is denoted by $v \xrightarrow{*} u$. A *cycle* is a *nonempty path* from a vertex to itself. A *self loop* is an edge $v \to v$. A *cutset* is a set of vertices which cuts all cycles in the graph. A *minimum cutset* is a cutset of minimum size.

A *rooted graph* is a triple (V,E,r) such that (V,E) is a directed graph, the *root* r is in V, and for any $v \in V$ there is a path $r \xrightarrow{*} v$. A *depth first search* (DFS) of a rooted graph is a way of exploring a rooted graph in linear time using an auxiliary stack. A detailed description of DFS can be found in Tarjan [8]. A DFS defines a partition of the edges into: 1) *Backward edges* - edges $v \to v'$ such that v' is already in the stack when v is pushed into the stack. 2) *Dag*[4] *edges* - all the other edges. The classification of edges may depend both on the graph and on the search order in the DFS. For a given graph $G=(V,E,r)$ and DFS α, we define the *DFS of G defined by* α, denoted by G^{α} to be the graph G where the edges are classified to dag and backward edges according to α.

Let $G=(V,E,r)$ be a rooted graph. A vertex v *dominates* a vertex v' if v cuts every path $r \xrightarrow{*} v'$. For the sake of clarity, note that according to this definition, every vertex v dominates itself. The set of vertices dominated by a vertex v is called *the dominated set of* v and is denoted by $D(v)$.

---

[4] In the literature the dag edges are classified further into *tree edges, forward edges* and *cross edges*. This finer classification will not be used here.

3. THE ALGORITHMS AND THE RELEVANT GRAPH CLASSES

3.1 Shamir-Rosen's Algorithm and the Class of Quasi-Reducible Graphs

In this section we describe Shamir-Rosen's algorithm and the classes of graphs related to it. For clarity of the presentation we refer to Shamir's algorithm (without Rosen's modification), called also Algorithm D, which *aborts* whenever the partial cutset found so far is not guaranteed to be a minimum one. The main data structures used by the algorithm are a stack and two types of markings given to the vertices: $n(v)$ which is the DFS number, and $l(v)$ which is used to identify the cutset vertices. The cutset vertices are collected into the output set S. The algorithm is taken from Shamir[1] and is given next:

ALGORITHM D.

```
1) Set $S \leftarrow \Phi$; set vertex counter $c \leftarrow 1$; clear all flags; push r (the root vertex) into the (empty) stack.
2) Set $n(top) \leftarrow c$, $c \leftarrow c+1$, $l(top) \leftarrow 0$.
3) If there is some unmarked edge $top \rightarrow v$, mark it and proceed, otherwise go to step 7.
4) If $v$ has not been visited so far, push $v$ into the stack and go to step 2.
5) If $top \rightarrow v$ is a backward edge, set $l(top) \leftarrow \max(l(top), n(v))$ and go to step 3.
6) Set $l(top) \leftarrow \max(l(top), l(v))$ and go to step 3.
7) If $l(top) = n(top)$, set $S \leftarrow S \bigcup \{top\}$ and $l(top) \leftarrow 0$.
7.5) If $l(top > n(top)$, print "graph is nonreducible" and abort.
8) Save the current top of the stack in $v'$; pop the stack; if the stack is empty halt; otherwise, set $l(top) \leftarrow \max(l(top), l(v'))$; go to step 3.
```

The labels $l(v)$ given by Algorithm D, change dynamically during the algorithm. In the sequel we use the notation *the termination value of a vertex* v. This is the value $l(v)$ given to v when Algorithm D backtracks from v, and is denoted by $l_{ter}(v)$.

A graph for which algorithm D finds a minimum cutset (i.e., the algorithm does not abort when applied to the graph) is called in by Rosen[2] a *quasi-reducible graph*. As noted by Rosen[2], the successful termination of algorithm D depends not only on the graph but also on the order by which the graph vertices are explored. This means that for a given graph G, algorithm D may successfully terminate when selecting a DFS α, and abort when selecting a DFS β. Due to this inconsistent behavior, a finer definition of the quasi-reducible graphs is required. A graph G is called *strongly quasi-reducible* if algorithm D successfully runs on G for *every* DFS chosen. A graph G is called *(weakly) quasi-reducible* if there *exists* a DFS α such that algorithm D successfully runs on G using α.

Let G be a graph, let α be a DFS defined on G and let G^α be the DFS of G defined by α. We say that algorithm D *successfully runs on* G^α if it successfully runs on G using the DFS α.

3.2 The Contraction Operations

Five contraction operations are used by the contraction algorithm:

1. IN0(v) - if vertex-indegree(v)=0 then contract G by removing v and all the edges leaving v.
2. OUT0(v) - the symmetric operation to IN0(v).
3. IN1(v) - if vertex-indegree(v)=1 and u ($u \neq v$) is the predecessor of v, then contract G, by *collapsing* v *into* u, as follows: For every edge $v \xrightarrow{e} v'$ remove e and add an edge $u \rightarrow v'$. Remove v and the edge $u \rightarrow v$. Remove all parallel edges created by this transformation

(See a graphic representation of this operation in Figure 1).

4. OUT1(v) - the symmetric operation to IN1(v).
5. LOOP(v) - if $v \rightarrow v$ is a self loop edge in G, then contract G by removing v and all the edges incident to v.

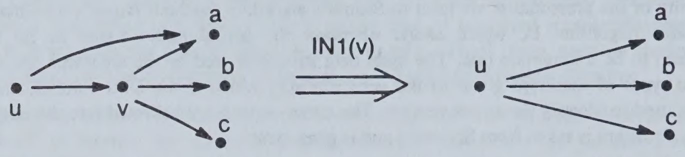


Figure 1: The IN1(v) contraction.

3.3 The Graph Classes

Six classes of graphs are of interest for our comparison:

1. Completely Contractible - graphs which can be completely contracted to an empty graph by the five contraction operations.
2. Backward Contractible - graphs which can be completely contracted to an empty graph or to a single vertex by the IN1 and LOOP operations.
3. Strongly Quasi-Reducible - graphs for which Shamir's algorithm guarantees finding a minimum cutset *independently* of the search order chosen.
4. (Weakly) quasi-Reducible - graphs for which *there exists* a search order by which Shamir's algorithm will find a minimum cutset for the graph.
5. Reducible - graphs which can be reduced to a single vertex using the IN1 operation and the operation of removing a self-loop edge.
6. Cyclically Reducible - graphs for which Wang-Lloyd-Soffa's algorithm finds minimum cutsets.

Recall that the three algorithms compared in this paper are related to the graph classes as follows: Shamir-Rosen's algorithm finds minimum cutsets for quasi-reducible graphs, Wang-Lloyd-Soffa's algorithm finds minimum cutsets for cyclically reducible graphs, and Levy-Low's algorithm finds minimum cutsets for completely contractible graphs.

4. QUASI-REDUCIBLE GRAPHS ARE BACKWARD CONTRACTIBLE

The main result of this paper, namely that the class of backward contractible graphs properly contains the class of quasi-reducible graphs, is derived in this section. To obtain this result it is required to show that if there exists a depth first search order according to which Shamir's algorithm finds a minimum cutset for a graph G, then it is possible to completely contract the graph by a series of IN1 and LOOP operations. The difficulty in proving this property is the different nature of the contraction algorithm and Shamir's algorithm. We overcome this difficulty using an inductive approach which uses the correctness of Shamir's algorithm itself in proving the inductive step. The approach is to show that if a graph G is quasi-reducible than it is possible to contract it by a *single contraction* (either IN1 or LOOP) to a smaller graph G' which is also quasi-reducible.

The structure of this analysis is as follows: First (Section 4.1) the induction basis is considered, where it is shown that every acyclic graph is both quasi-reducible and backward contractible. Second, the inductive step of the analysis is done in reverse; It is shown that if algorithm D successfully runs on a graph G, using a DFS α, then: 1) There exists a *specific vertex* where G can be contracted either by an IN1 contraction or by a LOOP contraction to a smaller graph G' (Section 4.2) and 2) There exists a DFS β, defined on G', such that algorithm D successfully runs on G'^{β} (Section 4.3). Lastly, we conclude that every quasi-reducible graph is backward contractible.

4.1 Acyclic Graphs -- The Induction Basis

LEMMA 1: Let $G=(V,E,r)$ be an acyclic directed graph rooted at r. Then:
1) G can be contracted by a series of IN1 contractions to a single vertex.
2) Algorithm D successfully runs on G without aborting, independently of the chosen search order α.

Proof: 1) Assume that (1) is false, and let G be the smallest graph violating (1). It is easy to see that if G is a *dag* rooted at r and if $|V|>1$, then there exists a vertex $v \neq r$ such that indegree(v)=1. Thus, it is possible to contract G to G' by IN1(v). Clearly, G' is a *dag* rooted at r; this contradicts the assumption that G is the smallest graph violating (1).
2) From (1), G is a reducible flow graph and the proof follows. ■

4.2 The Contraction Point

In this section we consider cyclic rooted graphs. For these graphs it is shown that there exists a specific vertex at which G can be contracted either by an IN1 contraction or by a LOOP contraction. In the following discussion (Section 4.2 - Section 4.3), let $G=(V,E,r)$ be a cyclic graph for which there exists a DFS α such that algorithm D will successfully run on G^{α}. Let S be the set of vertices selected by algorithm D to be in the cutset of G, and let $s \in S$ be the vertex with the highest DFS number among the vertices in S. Note that s is the last of all the vertices of S to enter the stack.

Next let us take a snapshot of the graph when s enters the stack. This snapshot is depicted in Figure 2. For simplicity it is assumed that the graph is searched bottom up and left to right. According to this snapshot we classify the vertices of the graph:

i. The set OLD contains the vertices from which the algorithm has already backtracked when s enters the stack.

ii. The set STACK contains all the vertices (not including s) that are in the stack when s joins the stack.

iii. The set $D(s)$ is the dominated set of s (the set includes s).

iv. The set NEW contains the rest of the vertices. Clearly, all the vertices in NEW are discovered after the discovery of s.

The graphs on which Algorithm D successfully runs, and the vertex s in particular, possess some important properties which are stated next in Lemma 2 and Lemma 3:

LEMMA 2: Let G^{α} be as defined above. Then,
1) If v is a vertex in the cutset (S) found by algorithm D then there exists at least one backward edge in G^{α} which enters v.
2) If $v \rightarrow u$ is a backward edge in G^{α} then there exists a vertex $t \in S$ such that $n(t) \geq n(u)$.

LEMMA 3: Let G^{α}, s and $D(s)$ be as defined above. Then,
1) There exists at least one backward edge in G^{α} that enters s.
2) If $v \rightarrow s$ is a backward edge entering s then $v \in D(s)$.
3) Let $v \neq s$ be a vertex dominated by s ($v \in D(s)-\{s\}$), and let $u \rightarrow v$ be an edge in G. Then,

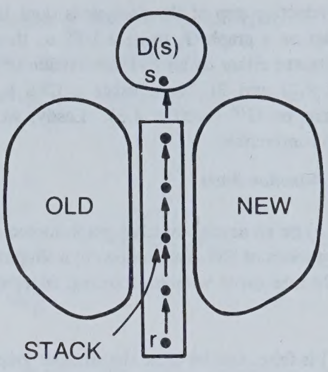


Figure 2: A snapshot of the graph when s enters the stack

$u \to v$ is a *dag* edge in G^{α} and $u \in D(s)$.

The proof of Lemma 2 follows directly from the behavior of algorithm D and from its correctness. The proof of Lemma 3 requires some details and is given in Levy [9]. From the properties stated in Lemma 2 and Lemma 3 it is now easy to show that there exists a vertex where G can be contracted:

THEOREM 4: Let G^{α}, s and $D(s)$ be as defined above. Then either of the following holds:
1) $|D(s)|=1$ and there exists a self loop $s \to s$.
2) $|D(s)|>1$ and there exists a vertex v such that $v \in D(s)-\{s\}$ and indegree(v)=1.

Proof: 1) If there exists only one vertex in $D(s)$, then Lemma 3 implies that there exists a backward edge $s \to s$. 2) If $|D(s)|>1$ let us remove s from the graph and look at $D(s)-s$. Since this subgraph is acyclic, and since every edge that enters a vertex in this subgraph must leave another vertex in this subgraph (Lemma 3), there exists a vertex v whose indegree in this subgraph is 0. Now return s to the graph, and we must have indegree(v)=1. ■

4.3 The Inductive Step

Having found the vertex at which G can be contracted to G', we next show that there exists a DFS β, defined on G', such that when algorithm D runs on G'^{β}, it will not abort. First, we deal with the IN1 contraction:

THEOREM 5: Let G, G^{α} and r be as defined above. Let $v \neq r$ be a vertex such that indegree(v)=1. Let G' be the graph produced from G by performing the contraction IN1(v). Then, there exists a DFS β defined on G', such that algorithm D will run on G'^{β} without aborting.

The theorem can be easily proved by observing that the edges in G' can be visited in the same relative order in which they are visited in G. A rigorous proof is given in Levy[9].

While the IN1 operation does not force a major change in the depth first search order, a LOOP operation causes more difficulties. The reason is that by removing edges in the LOOP operation, the order by which the vertices are visited by algorithm D, may be forced to change. Thus, it is not clear that algorithm D can still run successfully on the graph G' after performing an *arbitrary* LOOP operation. For this reason we need to concentrate on a LOOP operation which is performed on s.

Let G, G^α, s and r be as defined above, and let $|D(s)|=1$; let us also assume that s is not the root[5] (r). From Lemma 3 there exists in G a self loop $s \rightarrow s$, so we can contract G by LOOP(s). Let G' be the graph resulting from performing this contraction on G. The search order β defined on G' will be very similar to α. We enforce this similarity by the following rule: if $e_1, e_2, \cdots e_k$ are the edges leaving a vertex v in G, according to the search order α, then in β they will be searched in the same *relative* order (note, that only *relative* order can be kept since some of these edges may not exist in G').

Let R denote the application of algorithm D to G^α and R' denote the application of algorithm D to G'^β. In addition, let $n(v)$ and $n'(v)$ denote the DFS numbers given to v by R and R' respectively. Let $l(v)$ and $l'(v)$ be the labels given to v by R and R', and let $l_{ter}(v)$ and $l'_{ter}(v)$ be the values of v (under R and R') after backtracking from v. From the definition of β it is clear that until reaching the step of pushing s into the stack the behavior of R is identical to the behavior of R'. Thus for every vertex v in OLD, $l_{ter}(v)=l'_{ter}(v)$ and $n(v)=n'(v)$. For every vertex v in STACK, $n(v)=n'(v)$. Since R and R' behave identically until the step of pushing s to the stack, what remains to be shown is that R' will not abort after this step.

The following analysis (Lemma 6, Theorem 7 and Corollary 8) is done under the assumptions given above. First, Lemma 6 states that for every edge leaving a vertex in NEW, the edge classification in G^α is identical to the edge classification in G'^β.

LEMMA 6: Let v be a vertex in NEW. Then a) $v \rightarrow u$ is a backward edge in G^α if and only if it is a backward edge in G'^β, and b) If $v \rightarrow u$ is a backward edge then $u \in STACK$.

The correctness of the lemma is implied from the fact that NEW contains no cutset vertices. A rigorous proof may be found in Levy[9]. Next, based on Lemma 6, we show that the termination value given by R' to an arbitrary vertex, is identical to the termination value given to it by R.

THEOREM 7: Let v be a vertex which is backtracked by R' and which is in $STACK \bigcup NEW$. Then $l'_{ter}(v)=l_{ter}(v)$ and R' will not abort when backtracking from v.

Proof: Assume the contrary, and let v be the first vertex backtracked by R' for which the claim does not hold. Let $e_1, e_2, \cdots, e_k$ be the edges leaving v in G^α and let $v_1, v_2, \cdots, v_k$ be the head vertices of these edges, respectively. Let us now examine the edge e_i and the vertex v_i. Let $l_v(e_i)$ be the value returned to v when R traverses e_i, and let $l'_v(e_i)$ be the value returned to v when R' traverses e_i. Regardless of the set to which v_i belongs, we can show that $l_v(e_i)=l'_v(e_i)$:

1. If $v_i \in OLD$ then e_i is a *dag* edge both in G and in G'. Thus $l_v(e_i)=l'_v(e_i)$
2. If $v_i \in NEW$ then e_i is a *dag* edge and R' has successfully backtracked from v_i before returning the value $l'_v(e_i)$. Thus $l_v(e_i)=l_{ter}(v_i)$ and $l'_v(e_i)=l'_{ter}(v_i)$. Since v is the first vertex for which $l_{ter}(v) \neq l'_{ter}(v)$, then $l_{ter}(v_i)=l'_{ter}(v_i)$ and, thus, $l_v(e_i)=l'_v(e_i)$.
3. If $v_i \in STACK$ then e_i is a backward edge in G^α *iff* e_i is a backward edge in G'^β (if v is in

---

[5] The case where s is the root is trivial since the whole graph consists of one vertex and one self loop.

NEW it was proved in Lemma 6; if v is in STACK, it is trivial). If e_i is a backward edge then $l_v(e_i)=n(v_i)$ and $l'_v(e_i)=n'(v_i)$; also $v_i \in STACK$, so $n(v_i)=n'(v_i)$. Thus $l_v(e_i)=l'_v(e_i)$. If, on the other hand, e_i is a *dag* edge, then, as shown in (2) above, it must be true that $l_v(e_i)=l'_v(e_i)$.

4. If $v=s$, then e_i is a *dag* edge in G^α (Lemma 3), so $l_v(e_i)=0$. On the other hand, $l'_v(e_i)$ is not defined, since e_i is not in G'.

Next, let us examine the value computed for v before backtracking from v. From the observation above regarding the values returned to v when traversing e_i, it is obvious that $l(v)$ before backtracking v by R must be identical to $l'(v)$ before backtracking v by R'. Now, if $v \in NEW$, then $l(v)<n(s)$ (otherwise either R would abort here or v would be added to S) and therefore $l'(v)<n(s)$. In addition, $n'(v)\geq n(s)$. Thus, at backtrack time, $l'(v)<n'(v)$, so both R and R' backtrack from v without aborting and without changing its value, and the termination value must obey $l_{ter}(v)=l'_{ter}(v)$. On the other hand, if $v \in STACK$, then $n(v)=n'(v)$. Thus, in both applications, after backtracking v we must have $l_{ter}(v)=l'_{ter}(v)$. In addition, R' will not abort when backtracking v since R did not abort in this case.

By the contradiction, the claim is proved. ■

From Theorem 7, it immediately follows that if algorithm D successfully runs on G^α, it will also successfully run on G'^β. From Theorem 5 and Theorem 7, we conclude:

COROLLARY 8: Let $G=(V,E,r)$ be a cyclic directed graph rooted at r. If G is quasi-reducible, then it can be contracted either by an IN1 operation or by a LOOP operation to a rooted graph $G'=(E',V',r)$ which is quasi-reducible.

4.4 The Theorem

Finally, from Theorem 1 (the induction basis) and Corollary 8 (the inductive step) and by using inductive approach one can easily prove the main theorem of this section:

THEOREM 9: Let $G=(V,E,r)$ be a directed graph rooted at r. If G is quasi-reducible, then G is backward contractible.

5. THE HIERARCHY OF GRAPHS AND COMPARISON OF THE ALGORITHMS

Having proved Theorem 9, we can next summarize the relations among the graph classes discussed in this paper[6].

First, it is easy to show that the class of completely contractible graphs properly contains the class of backward contractible graphs.

Second, from Theorem 9 the class of backward contractible graphs contains the class of quasi-reducible graphs. Figure 3 depicts a graph which is backward contractible but not quasi-reducible (one may verify that this graph is not quasi-reducible by observing that any search of Algorithm D, starting at r, will end up with either $l_{ter}(b)=n(c)$ or $l_{ter}(b)=n(d)$, which will cause Algorithm D to abort when backtracking b). Thus, we may conclude that the inclusion proved in Theorem 9 is proper.

Third, by definition, the class of quasi-reducible graphs contains the class of strongly quasi-reducible graphs. In addition, as shown by Rosen[2], this inclusion is proper.

---

[6] Other relevant classes of graphs and the relations between them have been discussed by Lloyd[10].

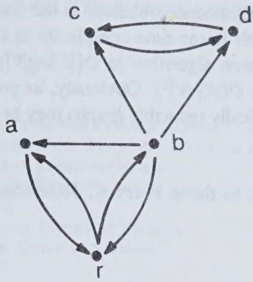


Figure 3: A backward contractible graph which is not quasi-reducible.

Fourth, the class of strongly quasi-reducible graphs was shown by Shamir[1] to contain the class of reducible graphs. Again, it is easy to show that this inclusion is proper.

Lastly, the result derived in Section 4 allows us to relate the class of cyclically reducible graphs to the other classes. This issue was addressed by Wang, Lloyd and Soffa [4] where it was shown that neither class contains the other but no other relation was found between the classes. In Lloyd, Soffa and Wang [5] it was shown that the class of cyclically reducible graphs is equivalent to the class of *forward reducible graphs*. These are graphs which can be completely contracted by the operations OUT0, OUT1 and LOOP. Using this result and the result derived in Section 4, it is now easy to compare cyclically reducible graphs to quasi-reducible graphs. Obviously, a class which is a "mirror image" of the class of forward reducible graphs (consisting of graphs which can be contracted using IN0, IN1, and LOOP) properly contains the class of quasi-reducible graphs. Thus, we may conclude that the class of cyclically reducible graphs is larger than the class of quasi-reducible graphs. In addition to this new result, it is obvious that the class of completely contractible graphs properly contains the class of cyclically reducible graphs.

Thus, we conclude that the first five classes can be organized in a complete hierarchy with proper inclusions: reducible $\subset$ strongly quasi-reducible $\subset$ quasi-reducible $\subset$ backward contractible $\subset$ completely contractible. In addition we have cyclically reducible $\subset$ complete contractible.

An interesting property of this hierarchy is that the classes of quasi-reducible graphs and strongly quasi-reducible graphs are "squeezed" between the classes of backward contractible graphs and reducible graphs. Recall that the backward contractible graphs and the reducible graphs are defined as graphs which can be contracted by certain operations. Also recall that the operation sets used in the definitions of these two classes are quite similar (IN1 and LOOP, versus IN1 and self-loop edge removal) and thus these classes can be assumed to have similar properties. This gives us a better understanding of the nature of the quasi-reducible graphs, which otherwise are defined in terms of a depth-first search algorithm.

Having constructed the hierarchy, it is now simple to compare the strength of the algorithm. If the performance objective is the number of graphs for which the algorithm finds a minimum cutset, it is obvious that the contraction algorithm is superior to the two other algorithms, and that Wang-Lloyd-Soffa's algorithm is superior to Shamir-Rosen's algorithm. As a matter of fact, the structure of the hierarchy suggests that the superiority of the contraction algorithm is quite significant, while the difference between the other two algorithms is smaller.

If, on the other hand, the performance objective is the cost of performing the algorithm, then Shamir-Rosen's algorithm with linear time complexity is the most efficient of the three. The cost of applying the contraction algorithm is $O(|E|\log|V|)$ and the cost of the original Wang-Lloyd-Soffa's algorithm[4] is $O(|E|\cdot|V|^2)$. Obviously, as pointed out by Lloyd, Soffa and Wang[5], minimum cutsets for cyclically reducible graphs may be found in $O(|E|\log|V|)$ time.

Acknowledgements: I would like to thank Barry K. Rosen for his helpful comments on this work.

REFERENCES

1. Shamir, A. "A Linear Time Algorithm for Finding Minimum Cutsets in Reducible Graphs," SIAM J. Computing, Vol. 8 No. 4 (1979) pp. 645-655.
2. Rosen, B. K. "Robust Linear Algorithms for cutsets", Journal of Algorithms 3(1982) pp. 205-212.
3. Wang, C.C., Lloyd, F.L. and Soffa, M.L. "Finding Minimum Decycling Sets for Cyclically Reducible Graphs", Technical Report 83-1, Dept. of Computer Science, Univ. of Pittsburgh, Pittsburgh, Pennsylvania.
4. Wang C.-C., Lloyd, F.L. and Soffa M.L. "Feedback Vertex Sets and Cyclically Reducible Graphs." *JACM*, Vol. 32, No. 2, pp. 296-313, April 1985.
5. Lloyd, F.L., Soffa, M.L. and Wang, C.-C. "Feedback Vertex Sets in Polynomial Time - a New Class." *Proc. 22nd Annual Allerton Conference on Communication, Control, and Computing, Oct. 3-5, 1984,* Allerton House, Monticello, IL, pp. 291-298.
6. Levy, H. and Low, D.W. "A New Algorithm for Finding Small Cycle Cutsets". G320-2721, IBM Los Angeles Scientific Center, Los Angeles, CA. June 1983.
7. Levy, H. and Low, D.W. "A Contraction Algorithm for Finding Small Cycle Cutsets" in Proceedings Eighteenth Annual Princeton Conf. on Inf. Sciences and Systems, Princeton, New Jersey. 1984.
8. Tarjan, R. E. "Depth First Search and Linear Graph Algorithms". SIAM J. Computing (1). 1972. pp. 146-160
9. Levy, H. "A Comparison of Low Complexity Algorithms for Finding Small Cycle Cutsets," G320-2758, IBM Los Angeles Scientific Center, Los Angeles, CA, June 1984.
10. Lloyd, F.L., Private Communication.

AN EFFICIENT ALGORITHM FOR FINDING A MAXIMUM CLIQUE OF AN OVERLAP GRAPH*

SUMIO MASUDA
Electrical Engineering Department and Systems Research Center
University of Maryland, College Park, Maryland 20742; on leave from
Department of Information and Computer Sciences
Osaka University, Toyonaka, Osaka 560, Japan
TOSHINOBU KASHIWABARA
Department of Information and Computer Sciences
Osaka University, Toyonaka, Osaka 560, Japan
KAZUO NAKAJIMA
Electrical Engineering Department, Institute for Advanced Computer Studies,
and Systems Research Center
University of Maryland, College Park, Maryland 20742
TOSHIO FUJISAWA
Department of Information and Computer Sciences
Osaka University, Toyonaka, Osaka 560, Japan

ABSTRACT

We present an efficient algorithm for finding a maximum clique of an overlap graph. When the graph is given in the form of a family of n intervals, the algorithm runs in $O(n \cdot \log n + Min\{m, n \cdot \omega\})$ time and with $O(n)$ space, where m is the number of edges, and ω is the cardinality of a maximum clique, respectively, of the graph.

1. INTRODUCTION

Let $F=\{I_1,I_2,...,I_n\}$ be a finite family of closed intervals on the real line $\mathbf{R}$. For two distinct intervals I_j and I_k in F, we say that I_j and I_k *overlap* with each other if they intersect with each other but neither one of them contains the other. A graph $G=(V,E)$ is called an *overlap graph* for F if there is a one-to-one correspondence between V and F such that two vertices in V are adjacent to each other if and only if the corresponding two intervals in F overlap with each other. We denote by $G(F)=(V_F,E_F)$ the overlap graph for F. It is known that any overlap graph is a circle graph, which is an intersection graph defined for a family of chords of a circle, and vice versa [6].

Let $G=(V,E)$ be a graph. A subset S of V is called a *clique* of G if any two distinct vertices in S are adjacent to each other. A *maximum clique* of G is a clique whose cardinality is the largest among all cliques of G. The problem of finding a maximum clique of G is, in general, NP-hard [1,4]. However, it is solvable in polynomial time when G is restricted to such a graph as a chordal graph [5], comparability graph [7], or circular-arc graph [8]. Furthermore, in the case of a circle graph, namely, an overlap graph, several polynomial time algorithms [2,6,8,10] have been proposed for finding a maximum clique of G under the assumption that the graph is given in the form of a family of intervals F.

The first such algorithm was due to Gavril [6], and it runs in $O(n^3)$ time, where $n=|F|=|V_F|$. Later, Rotem and Urrutia [10] and Buckingham [2] proposed more efficient

---

* This work was supported in part by National Science Foundation Grants DMC-84-51510 and CDR-85-00108, and in part by a grant from AT&T Information Systems.

algorithms whose time complexities are $O(n^2)$ and $O(n \cdot \log n + m \cdot \log\omega)$, respectively, where $m = |E_F|$ and ω is the cardinality of a maximum clique. Recently, Hsu [8] developed an algorithm for finding a maximum weight clique of $G(F)$ for the case when each interval in F is assigned an integer as its weight. Although he claimed that his algorithm runs in $O(n^2 + m \cdot \log\log n)$ time, we can give an $O(n \cdot \log n + m \cdot \log\log n)$ time implementation of the algorithm by using Buckingham's algorithm [2] for generating the edges of $G(F)$ in the preprocessing stage.

In this paper, we present a more efficient algorithm for finding a maximum clique of an overlap graph when the graph is given in the form of a family of n intervals. Using efficient data manipulation on longest increasing subsequences of a sequence of numbers [3], we will show that the algorithm can be implemented to run in $O(n \cdot \log n + Min\{m, n \cdot \omega\})$ time and with $O(n)$ space. (Because of the space limit, most proofs are omitted. They may be found in Ref.[9].)

2. PRELIMINARIES

Let $F = \{I_1, I_2, \ldots, I_n\}$ be a finite family of closed intervals on the real line $\mathbf{R}$. For $j = 1,2,\ldots,n$, let l_j (resp., r_j) denote the coordinate of the left (resp., right) endpoint of interval I_j. In other words, $I_j = [l_j, r_j]$ for $j = 1,2,\ldots,n$. For two distinct intervals I_j and I_k in F, we say that I_j *contains* I_k if $l_j \le l_k \le r_k \le r_j$, and that I_j and I_k *intersect* with each other if there exists some real number c such that $l_j \le c \le r_j$ and $l_k \le c \le r_k$. Furthermore, if I_j and I_k intersect with each other but neither one of them contains the other, in other words, if $l_j < l_k \le r_j < r_k$ or $l_k < l_j \le r_k < l_j$, then we say that I_j and I_k *overlap* with each other.

The overlap graph for F, denoted by $G(F)$, is defined as:

$G(F) = (V_F, E_F)$,

where $V_F = \{v_1, v_2, \ldots, v_n\}$,

and $E_F = \{(v_j, v_k) \mid I_j$ and I_k overlap with each other$\}$.

As an example, a family of intervals and its corresponding overlap graph are shown in Fig. 1, where the intervals are drawn as distinct horizontal line segments for clarity.

A family of n intervals F is said to be *canonical* if the coordinates of the endpoints of the intervals are all distinct integers between 1 and $2n$.

Remark 1[9]. When F is not canonical, one can construct a canonical family of intervals F' such that $G(F) = G(F')$ in $O(n \cdot \log n)$ time and with $O(n)$ space. ☐

We now introduce the concept of a longest increasing subsequence [3] of a sequence of numbers. This plays an important role in the development of our efficient algorithm for finding a maximum clique of an overlap graph.

Let $\pi = [\pi(1), \pi(2), \ldots, \pi(s)]$ be a sequence of numbers. For $i = 1,2,\ldots,s$, we say that $\pi(i)$ is an element of π and this is denoted by $\pi(i) \in \pi$. The number, s, of the elements of π is called the *length* of π, and is denoted by $|\pi|$. π is said to be *simple* if $\pi(i) \ne \pi(j)$ for any two integers i and j such that $1 \le i \ne j \le s$. In the following, a sequence means a simple sequence of numbers.

For a sequence $\pi = [\pi(1), \pi(2), \ldots, \pi(s)]$ and an element x of π, let $pos(x)$ denote the position of x in π, that is, $pos(\pi(i)) = i$ for $i = 1,2,\ldots,s$. Note that the value of $pos(x)$ is uniquely determined for any element x of π, since π is simple.

A subsequence $Y = [\pi(i_1), \pi(i_2), \ldots, \pi(i_t)]$ is called an *increasing subsequence* (abbreviated to an IS) of π if Y satisfies the following two conditions:

(1) $1 \le i_1 < i_2 < \cdots < i_t \le s$.

(2) $\pi(i_1) < \pi(i_2) < \cdots < \pi(i_t)$.

A *longest increasing subsequence* (abbreviated to an LIS) of π is an IS whose length is the maximum among all IS's of π.

For $i = 1,2,\ldots,s$, let $\lambda(\pi,\pi(i))$ be defined as follows:

$$\lambda(\pi,\pi(i)) \triangleq Max\,\{|Y| \mid Y \text{ is an IS of } \pi \text{ and } Y \text{ contains } \pi(i) \text{ as its first element}\}.$$

For example, a sequence $\pi = [2,4,3,5,1]$ has two LIS's, $[2,3,5]$ and $[2,4,5]$. For this sequence, $\lambda(\pi,1)=1$, $\lambda(\pi,2)=3$, $\lambda(\pi,3)=2$, $\lambda(\pi,4)=2$ and $\lambda(\pi,5)=1$.

Let t be the length of an LIS of π. Let an ordered set $Q(\pi)$ be defined as $Q(\pi) \triangleq \langle X_1, X_2, \ldots, X_t \rangle$, where $X_i = \{x \in \pi \mid \lambda(\pi,x) = i\}$ for $i = 1,2,\ldots,t$. We call $Q(\pi)$ the *λ-representation* of π. For example, the λ-representation of $[2,4,3,5,1]$ is $\langle \{5,1\},\{4,3\},\{2\} \rangle$.

Let π be an arbitrary sequence, and suppose that $Q(\pi) = \langle X_1, X_2, \ldots, X_t \rangle$. Then, the following five lemmas hold for π and $Q(\pi)$.

Lemma 1. Let k be an integer such that $1 \le k \le t$, and let x and y be any two elements in X_k such that $x < y$. Then, $pos(x) > pos(y)$. □

Lemma 2. Let k be an integer such that $1 \le k \le t$, and let x_1 be any element in X_k. Let $Y = [x_1, x_2, \ldots, x_k]$ be an IS of length k which contains x_1 as its first element. Then, $x_i \in X_{k-i+1}$ for $i = 2,3,\ldots,k$. □

Corollary 1. For $j = 1,2,\ldots,t$, $X_j \ne \phi$. □

Lemma 3. Let x be the smallest number among the elements of π. Let X_k be the set in $Q(\pi)$ which contains x . (Such a set X_k is uniquely determined since π is simple.) Then if $k \ne t$, $|X_k| \ge 2$. □

Lemma 4. Let k be an integer such that $2 \le k \le t$, and let x_1 be an element in X_k. Let $y = Min\,\{z \mid z \in X_{k-1} \text{ and } z > x_1\}$. Then, there exists an IS, Y of π such that $|Y| = k$, $y \in Y$ and Y contains x_1 as its first element. □

Lemma 5. Let k be an integer such that $1 \le k \le t$. For two elements x_1 and y_1 in X_k, let $[x_1, x_2, \ldots, x_k]$ and $[y_1, y_2, \ldots, y_k]$ be any IS's of π. Let $z_i = Min\,\{x_i, y_i\}$ for $i = 1,2,\ldots,k$. Then, $[z_1, z_2, \ldots, z_k]$ is an IS of π. □

3. MAXIMUM CLIQUES AND LONGEST INCREASING SUBSEQUENCES

Let $F = \{I_1, I_2, \ldots, I_n\}$ be a canonical family of intervals. Without loss of generality, we can assume that $r_1 < r_2 < \cdots < r_n$. (The renumbering of the indices can be performed in $O(n)$ time by using bucket sort [1], if necessary.)

For $i = 1,2,\ldots,2n$, we define a set of intervals S_i as follows:

$$S_i \triangleq \{I_j \in F \mid l_j < i + 0.5 < r_j\}.$$

Let τ_i be the sequence of the indices of the intervals of S_i sorted in the ascending order of the coordinates of their left endpoints. For example, for the canonical family of intervals shown in Fig. 2, $S_4 = \{I_1, I_2, I_3, I_4\}$, $\tau_4 = [2,1,3,4]$, and $S_5 = \{I_2, I_3, I_4\}$, $\tau_5 = [2,3,4]$.

In the following, G_i is used to denote the overlap graph $G(S_i)$, which is an induced subgraph of $G(F)$. And $\omega(H)$ is used to denote the cardinality of a maximum clique of a graph H. The following two theorems hold.

Theorem 1. Let h be an integer such that $\omega(G_h) = Max\,\{\omega(G_i) \mid i = 1,2,\ldots,2n\}$. Then, a maximum clique of G_h is also a maximum clique of $G(F)$. □

Theorem 2. Let i be an integer such that $1 \leq i \leq 2n$. Then, the maximum cliques of G_i and the LIS's of τ_i are in one-to-one correspondence. □

Hereafter, we use the shorthand notation $\omega(F)$ to denote $\omega(G(F))$. Then the following corollary is an immediate consequence of Theorem 1.

Corollary 2. $\omega(F)=Max\,\{\omega(G_i) \mid i=1,2,\dots,2n\}$. □

Based on Theorem 1, one can find a maximum clique of $G(F)$ by performing the following steps (I) and (II).

(I) Find an integer h such that $\omega(G_h)=Max\,\{\omega(G_i) \mid i=1,2,\dots,2n\}(=\omega(F))$.

(II) Find a maximum clique of graph G_h.

From Theorem 2, Step (I) can be performed by comparing the lengths of the LIS's of the sequences $\tau_1, \tau_2, \dots, \tau_{2n}$. In the past, Fredman [3] presented an $O(|\pi| \cdot \log |\pi|)$ time algorithm for finding the length of an LIS of a sequence π. By executing his algorithm $2n$ times, one can complete Step (I). However, such a method requires $O(n^2 \cdot \log n)$ time.

Since the length of an LIS of τ_i is equal to $|Q(\tau_i)|$ for $i=1,2,\dots,2n$, it is sufficient to find $Q(\tau_1), Q(\tau_2), \dots, Q(\tau_{2n})$ to carry out Step (I). It is easy to obtain $Q(\tau_1)$. Furthermore, one can efficiently generate $Q(\tau_{i+1})$ from $Q(\tau_i)$ for $i=1,2,\dots,2n-1$ by utilizing the following relationships between τ_i and τ_{i+1}, which are stated in Remarks 2 and 3. The details of such generations will be described later.

Let $p(k)$ denote the endpoint whose coordinate is k for $k=1,2,\dots,2n$. Let I_j be the interval which contains $p(i+1)$ as one of its endpoints.

Remark 2. If $p(i+1)$ is the left endpoint of I_j, then τ_{i+1} is obtained by appending j at the end of τ_i, as can be seen in Fig.3(a). □

Remark 3. If $p(i+1)$ is the right endpoint of I_j, then j is the smallest element of τ_i, and τ_{i+1} is obtained by deleting j from τ_i, as can be seen in Fig.3(b). □

For an integer i such that $1 \leq i \leq 2n-1$, let $Q(\tau_i) = < X_1, X_2, \dots, X_t >$ and $Q(\tau_{i+1}) = < X'_1, X'_2, \dots, X'_u >$. For each element x in τ_i, we use $b(x)$ to denote $\lambda(\tau_i, x)$, that is, $x \in X_{b(x)}$. In the following, we show four theorems which give the relationships between $Q(\tau_i)$ and $Q(\tau_{i+1})$.

Theorem 3. Suppose that $p(i+1)$ is the right endpoint of interval I_j. If $|X_{b(j)}| \neq 1$ or $b(j) \neq t$, then $u=t$, $X'_{b(j)} = X_{b(j)} - \{j\}$, and $X'_k = X_k$ for $k=1,2,\dots,b(j)-1, b(j)+1, b(j)+2, \dots, t$. On the other hand, if $b(j)=t$ and $|X_t|=1$, then $u=t-1$ and $X'_k = X_k$ for $k=1,2,\dots,t-1$.

Proof. As was mentioned in Remark 3, j is the smallest element of τ_i and τ_{i+1} is obtained by deleting j from τ_i. This implies that $\lambda(\tau_i, x) = \lambda(\tau_{i+1}, x)$ for any element x of τ_{i+1}. Therefore, the theorem holds. □

Remark 4. In Theorem 3, if $b(j) \neq t$, then $|X_{b(j)}| \geq 2$ from Lemma 3. Therefore, each set X'_k in $Q(\tau_{i+1})$ is not empty. □

Lemma 6. Suppose that $p(i+1)$ is the left endpoint of interval I_j. Let x be an element of τ_i. Then, the following inequalities hold:

$$\lambda(\tau_i, x) \leq \lambda(\tau_{i+1}, x) \leq \lambda(\tau_i, x)+1.$$

Proof: It is clear from Remark 2 that any IS of τ_i is an IS of τ_{i+1}. Therefore, $\lambda(\tau_i, x) \leq \lambda(\tau_{i+1}, x)$.

Let Y be a longest IS among the IS's of τ_{i+1} which contain x as its first element. If Y does not contain j, Y is also an IS of τ_i. If Y contains j, the sequence obtained by deleting j from Y is an IS of τ_i. Therefore, $\lambda(\tau_{i+1}, x) \leq \lambda(\tau_i, x)+1$. □

Corollary 3. In the case in which $p(i+1)$ is the left endpoint of interval I_j, $t \leq u \leq t+1$. □

In the case in which $p(i+1)$ is the left endpoint of I_j, we define sets of indices A_k's for $k=1,2,\ldots,u+1$ as follows:

$$A_1 \triangleq \{j\},$$

$$A_k \triangleq \{x \mid \lambda(\tau_{i+1},x)=k \text{ and } \lambda(\tau_i,x)=k-1\} = X_k' \cap X_{k-1}, \quad k=2,3,\ldots,u$$

$$A_{u+1} \triangleq \phi.$$

From Lemma 6, we can immediately obtain the following theorem.

Theorem 4. For $k=1,2,\ldots,u$,

$$X_k' = (X_k - A_{k+1}) \cup A_k = (X_k \cup A_k) - A_{k+1},$$

where X_{t+1} is assumed to be empty when $u=t+1$. □

Theorem 4 implies that, if we can obtain $A_1, A_2, \ldots, A_{t+1}$, we can update the λ-representation. The following two theorem play an important role in achieving this update efficiently.

Theorem 5. Suppose that $p(i+1)$ is the left endpoint of interval I_j. If $A_k=\phi$ for some integer k such that $2 \leq k \leq t$, then $A_{k+1}=A_{k+2}=\cdots=A_{t+1}=\phi$.

Proof : Assume that $A_k=\phi$ and $A_{k+1} \neq \phi$. Let x_1 be an element of A_{k+1}. From the definition of A_{k+1}, $x_1 \in X_{k+1}' \cap X_k$. Let $[x_1, x_2, \ldots, x_{k+1}]$ be an IS of τ_{i+1}. Then $x_2 \in X_k'$ from Lemma 2. Furthermore, since $x_1 \in X_k$, $x_{k+1}=j$ from Remark 2, and hence $[x_1, x_2, \ldots, x_k]$ is an IS of τ_i. This implies by Lemma 2 that $x_2 \in X_{k-1}$. Therefore, $x_2 \in X_k' \cap X_{k-1} = A_k$, which contradicts the assumption that $A_k=\phi$.

By repeatedly using this argument, the proof will be completed. □

Theorem 6. Suppose that $p(i+1)$ is the left endpoint of interval I_j. Then, for $k=2,3,\ldots,t+1$, $A_k=\{x \in X_{k-1} \mid x < Max(A_{k-1})\}$ if $A_{k-1} \neq \phi$.

Proof. Let k be an integer such that $2 \leq k \leq t+1$.

("$\subseteq$" *part*) Let x_1 be an element of A_k. From the definition of A_k, $x_1 \in X_k'$ and $x_1 \in X_{k-1}$. Let $[x_1, x_2, \ldots, x_k]$ be an IS of τ_{i+1}. From Lemma 2, $x_2 \in X_{k-1}'$. Therefore, $x_2 \in (X_{k-1} - A_k) \cup A_{k-1}$ due to Theorem 4.

Now assume that $x_1 \geq Max(A_{k-1})$. Since $x_1 < x_2$, $x_2 \notin A_{k-1}$. Hence, $x_2 \in X_{k-1} - A_k \subseteq X_{k-1}$. This implies that there exists an IS, Y of τ_i of length $k-1$ which contains x_2 as its first element. Clearly, placing x_1 in front of Y yields an IS of τ_i of length k. This contradicts the fact that $x_1 \in X_{k-1}$. Therefore, $x_1 < Max(A_{k-1})$, and hence $x_1 \in \{x \in X_{k-1} \mid x < Max(A_{k-1})\}$.

("$\supseteq$" *part*) Let x_1 be an element of $\{x \in X_{k-1} \mid x < Max(A_{k-1})\}$. Let $y_1 = Max(A_{k-1})$. There are two cases to be considered separately.

Case 1: $k=2$.

Since $A_1=\{j\}$, $x_1 < j$. Furthermore, $pos(x_1) < pos(j)$ in τ_{i+1} since j is the last element of τ_{i+1}. Thus, $[x_1, j]$ is an IS of τ_{i+1}, which implies that $\lambda(\tau_{i+1}, x_1) \geq 2$. Since $x_1 \in X_1$, $x_1 \in X_2'$ from Lemma 6, and hence $x_1 \in X_1 \cap X_2' = A_2$.

Case 2: $k \geq 3$.

Since $y_1 \in A_{k-1}$, $y_1 \in X_{k-1}' \cap X_{k-2}$. Let $Y=[y_1, y_2, \ldots, y_{k-1}]$ be an IS of τ_{i+1}. Since $y_1 \in X_{k-2}$, $y_{k-1}=j$. Thus, $[y_1, y_2, \ldots, y_{k-2}]$ is an IS of τ_i. Furthermore, since $x_1 \in X_{k-1}$, there exists an IS $[x_1, x_2, \ldots, x_{k-1}]$ of τ_i. Therefore, from Lemma 2, x_q, $y_{q-1} \in X_{k-q}$ for $q=2,3,\ldots,k-1$. Let $z_q \triangleq Min\{x_{q+1}, y_q\}$ for $q=1,2,\ldots,k-2$. Then, $[z_1, z_2, \ldots, z_{k-2}]$ is an IS of τ_i from Lemma 5.

Since $y_{k-2}<y_{k-1}=j$, $z_{k-2}<j$. Therefore, $[z_1,z_2,\ldots,z_{k-2},j]$ is an IS of τ_{i+1}.

Suppose that $pos(x_1)<pos(y_1)$ in τ_i. Since $pos(x_1)<pos(x_2)$, $pos(x_1)<pos(z_1)$. On the other hand, suppose that $pos(x_1)>pos(y_1)$ in τ_i. Then $pos(y_1)<pos(x_2)$. Since $y_1,x_2 \in X_{k-1}$, $x_2<y_1$ from Lemma 1. This implies that $z_1=x_2$. Consequently, in both cases, $pos(x_1)<pos(z_1)$ in τ_i. Furthermore, it is clear that $x_1<z_1$ since $x_1<y_1$ and $x_1<x_2$. Therefore, $[x_1,z_1,z_2,\ldots,z_{k-2}]$ is an IS of τ_i, and hence $[x_1,z_1,z_2,\ldots,z_{k-2},j]$ is an IS of τ_{i+1}. This implies that $x_1 \in X_k'$. Since $x_1 \in X_{k-1}$, $x_1 \in X_k' \cup X_{k-1}=A_k$. □

4. ALGORITHM FOR FINDING A MAXIMUM CLIQUE

Based on Theorems 3, 4, 5 and 6, one can find $Q(\tau_i)$ for $i=1,2,\ldots,2n$. Our algorithm efficiently generates all $Q(\tau_i)$'s and performs the execution of Step (I) which was described in the preceding section.

In this section we assume that a given family of intervals F is canonical and that it is represented by two lists L_F^1 and L_F^2, where, for $i=1,2,\ldots,2n$, the contents of the i-th cells of L_F^1 and L_F^2 are defined as follows:

(1) $L_F^1(i)=j$ if and only if $p(i)$ is one of the endpoints of interval $I_j \in F$,

(2) $L_F^2(i)=$"l" (resp., "r") if and only if $p(i)$ is the left (resp., right) endpoint of $I_{L_F^1(i)}$.

For example, the family of intervals given in Fig. 2 is represented by the two lists shown in Fig. 4. If F is given in some other form, one can easily construct L_F^1 and L_F^2 in $O(n)$ time by using bucket sort [1] so long as F is canonical. A formal description of our algorithm, called Algorithm 1, is given in Fig.5 on the following page.

We now briefly explain how Algorithm 1 works.

For $i=2,3,\ldots,2n$, $Q(\tau_i)$ is represented as the ordered set $<X_1,X_2,\cdots,X_{LAST}>$ after the $(i-1)$-th execution of the statements between lines 3 and 23. And, for each element y of τ_i, $b(y)$ is determined in such a way that $b(y)=\lambda(\tau_i,y)$.

Based on Theorems 4, 5 and 6, the statements from lines 3 to 19 are executed for the case when $p(i)$ is the left endpoint. For each integer k, INS_k and DEL_k correspond to A_k and A_{k+1}, respectively. According to Theorem 6, DEL_k is determined as $\{x \in X_k \mid x<Max(INS_k)\}$ at line 17 if $INS_k=DEL_{k-1}\neq\phi$. From Theorem 4, X_k should be updated to be $(X_k - A_{k+1}) \cup A_k$. Therefore, while $DEL_k\neq\phi$, the statement "$X_k \leftarrow (X_k - DEL_k) \cup INS_k$" is executed at line 14. When $DEL_k=\phi$, the statement "$X_k \leftarrow X_k \cup INS_k$" is executed at line 18. In this case, $X_{k+1},X_{k+2},\ldots,X_{LAST}$ are not updated due to Theorem 5, and the i-th execution of the outermost **for** loop starts immediately.

For the case when $p(i)$ is the right endpoint, the statements from lines 20 to 23 are executed according to Theorem 3.

In Algorithm 1, for $i=1,2,\ldots,2n$, integer h is determined in such a way that $\omega(G_h)=Max\{\omega(G_q) \mid q=1,2,\ldots,i\}$ and the value of $\omega(G_h)$ is stored in variable POS. The values of h and POS are updated at line 13 only if all the conditions at the 3rd, 9th and 12th lines are satisfied, that is, only if i satisfies the conditions that $|Q(\tau_i)|=|Q(\tau_{i-1})|+1$ and $|Q(\tau_i)|>POS$.

The correctness of Algorithm 1 follows from Theorems 3, 4, 5 and 6. The evaluation of the complexities of Algorithm 1 is given below.

Algorithm 1.

Input : A canonical family of intervals $F=\{I_1,I_2,...,I_n\}$ represented by two lists L_F^1 and L_F^2.
Output : An integer h such that $\omega(G_h)=Max\{\omega(G_i) \mid i=1,2,...,2n\}$.
Method :

```
begin
  $X_1 \leftarrow \{L_F^1(1)\}$; $b(L_F^1(1)) \leftarrow 1$; $LAST \leftarrow 1$; $h \leftarrow 1$; $POS \leftarrow 1$;
  for $i \leftarrow 2$ until $2n$ do
     begin
        if $L_F^2(i)=$"$l$" then
        if $X_1=\phi$
           then
              begin
                 $X_1 \leftarrow \{L_F^1(i)\}$; $b(L_F^1(i)) \leftarrow 1$; $LAST \leftarrow 1$
              end
           else
              begin
                 $DEL_1 \leftarrow \{x \in X_1 \mid x < L_F^1(i)\}$;
                 $k \leftarrow 1$; $INS_1 \leftarrow \{L_F^1(i)\}$;
                 while $DEL_k \neq \phi$ do
                    begin
                       if $k = LAST$ then
                          begin
                             $LAST \leftarrow LAST+1$;
                             $X_{LAST} \leftarrow \phi$;
                             if $LAST > POS$ then
                                begin
                                   $h \leftarrow i$; $POS \leftarrow LAST$
                                end
                          end;
                       $X_k \leftarrow (X_k - DEL_k) \cup INS_k$;
                       for each $x \in INS_k$ do $b(x) \leftarrow k$;
                       $INS_{k+1} \leftarrow DEL_k$; $M \leftarrow Max(INS_{k+1})$;
                       $DEL_{k+1} \leftarrow \{x \in X_{k+1} \mid x < M\}$; $k \leftarrow k+1$
                    end;
                 $X_k \leftarrow X_k \cup INS_k$;
                 for each $x \in INS_k$ do $b(x) \leftarrow k$
              end;
        if $L_F^2(i)=$"$r$" then
        if $b(L_F^1(i))=LAST$ and $|X_{LAST}|=1$
        then $LAST \leftarrow LAST-1$
        else $X_{b(L_F^1(i))} \leftarrow X_{b(L_F^1(i))} - \{L_F^1(i)\}$
     end
  Output $h$
end □
```

Fig.5 Description of Algorithm 1.

In order to efficiently implement Algorithm 1, we use the following data structures to manipulate sets X_k's for $k=1,2,\ldots,LAST$.

1) $LIST_k$: A list of the elements of X_k sorted in the ascending order of their values. Each element x in $LIST_k$ has a pointer $NEXT(x)$ to the next element in $LIST_k$.
2) $POINT_k$: A pointer to the first cell of $LIST_k$.

For example, the data structures for the λ-representation $<\{5,8\},\{1,4,7\},\{2,3\}>$ are illustrated in Fig.6.

Since $|X_1|+|X_2|+\cdots+|X_{LAST}|\leq n$ at any time, the space complexity of Algorithm 1 is $O(n)$. In order to evaluate the time complexity of Algorithm 1, we explain how to perform the set manipulations at lines 14, 17 and 23.

1) Line 14: Let m_1 and m_2 be the smallest and largest elements in INS_k, respectively, and let m_3 be the largest element in DEL_k. (These elements can be determined when we find DEL_k and INS_k.) In order to execute the statement "$X_k \leftarrow (X_k - DEL_k) \cup INS_k$" at line 14, we need to make two assignments, "$POINT_k \leftarrow m_1$" and "$NEXT(m_2) \leftarrow NEXT(m_3)$". These operations require $O(1)$ time. Similarly, the statement at line 18 can be executed in $O(1)$ time.
2) Line 17: All we need to do is to visit every element of $LIST_{k+1}$ by tracing the pointers since the elements are already sorted. Therefore, each execution of this line can be done in $O(|DEL_{k+1}|+1)$ time. Similarly, the operation at line 6 can be performed in $O(|DEL_1|+1)$ time.
3) Line 23: The statement "$X_{b(L_F^1(i))} \leftarrow X_{b(L_F^1(i))} - \{L_F^1(i)\}$" can be executed by the simple assignment operation "$POINT_{b(L_F^1(i))} \leftarrow NEXT(L_F^1(i))$". This assignment requires a constant time for each execution.

Consequently, each execution of lines 6, 15, 16, 17 and 19 takes $O(|DEL_1|+1)$, $O(|INS_k|)$, $O(|DEL_k|)$, $O(|DEL_{k+1}|+1)$, and $O(|INS_k|)$ time, respectively. Each of the other lines requires only $O(1)$ time for its execution.

For $j=1,2,\ldots,n$, let $q_1(j)$ (resp., $q_1'(j)$) be the number of times in which j is added to one of the sets DEL_k's (resp., INS_k's) during the execution of Algorithm 1. Let $SUM \triangleq q_1(1)+q_1(2)+\cdots+q_1(n)$. Then, clearly $q_1'(1)+q_1'(2)+\cdots+q_1'(n)=n+SUM$. Therefore, the total sum of the execution times of lines 6, 15, 16, 17 and 19 is $O(n+SUM)$. Furthermore, lines 8 to 14 each are executed at most $O(SUM)$ times throughout the entire execution of the algorithm. Since the outermost **for** loop is executed $(2n-1)$ times, the remaining lines require $O(n)$ time to execute. Consequently, the time complexity of Algorithm 1 is $O(n+SUM)$.

Let $q_2(j)$ be defined as the subscript of the set containing j in the λ-representation $Q(\tau_{r_j-1})$, that is, $q_2(j)=\lambda(\tau_{r_j-1},j)$. Clearly $q_2(j)=q_1'(j)$. Thus, $SUM<q_2(1)+q_2(2)+\cdots+q_2(n)$. Since there exists an IS, Y of τ_{r_j-1} of length $q_2(j)$ which contains j as its first element, graph G_{r_j-1} has a clique Y' such that $|Y'|=q_2(j)$ and $v_j \in Y'$. Therefore, $q_2(j)$ is less than or equal to the degree of v_j in $G(F)$ for $j=1,2,\ldots,n$, and hence $SUM<2\cdot|E_F|$. Furthermore, it is clear that $q_2(j)=\lambda(\tau_{r_j-1},j)\leq\omega(F)$, which implies that $SUM<n\cdot\omega(F)$. Consequently, we have the following theorem.

Theorem 7. The time and space complexities of Algorithm 1 are $O(n+Min\{|E_F|, n\cdot\omega(F)\})$ and $O(n)$, respectively. □

By executing Algorithm 1, one can obtain such an integer h that $\omega(G_h)=Max\{\omega(G_i) \mid i=1,2,\ldots,2n\}=\omega(F)$. Due to Theorems 1 and 2, a maximum clique of $G(F)$ can be found by computing an LIS of sequence τ_h.

Suppose that Algorithm 1 is applied to F once again. Then, $Q(\tau_h)$ is obtained as the ordered set $<X_1, X_2, \ldots, X_{LAST}>$ after the $(h-1)$-th execution of the outermost **for** loop. We can obtain an LIS of τ_h by applying the following procedure to such an ordered set.

Procedure 1.

```
begin
    $x_{LAST} \leftarrow$ any element of $X_{LAST}$;
    for $k \leftarrow LAST-1$ until 1 step -1 do
        $x_k \leftarrow Min\ \{y \in X_k \mid y > x_{k+1}\}$;
    Output $[x_{LAST}, x_{LAST-1}, \ldots, x_1]$
end □
```

Lemma 7. Procedure 1 correctly finds an LIS of τ_h in $O(n)$ time.

Proof. The repeated application of Lemma 4 proves the correctness of the procedure. Since $|X_1| + |X_2| + \cdots + |X_{LAST}| = |S_h| \leq n$, the time complexity is clearly $O(n)$. □

Theorem 7 and Lemma 7 immediately lead to the following theorem.

Theorem 8. Given a canonical family of n intervals F, a maximum clique of the overlap graph $G(F)=(V_F, E_F)$ can be found in $O(n + Min\ \{|E_F|,\ n \cdot \omega(F)\})$ time and with $O(n)$ space. □

By combining this theorem and Remark 1, we have the following theorem.

Theorem 9. For any family of n intervals F, a maximum clique of $G(F)=(V_F, E_F)$ can be found in $O(n \cdot \log n + Min\ \{|E_F|,\ n \cdot \omega(F)\})$ time and with $O(n)$ space. □

REFERENCES

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, *The Design and Analysis of Computer Algorithms,* Addison-Wesley, Reading, MA, 1974.

[2] M.A. Buckingham, "Circle graphs", Technical Report NSO-21, Courant Institute of Mathematical Sciences, New York University, New York, NY, 1980.

[3] M.L. Fredman, "On Computing the Length of the Longest Increasing Subsequences", *Discrete Mathematics,* Vol. 11, pp. 29-35, 1975.

[4] M.R. Garey and D.S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-Completeness,* W.H. Freeman and Co., San Francisco, CA, 1979.

[5] F. Gavril, "Algorithms for Minimum Coloring, Maximum Clique, Minimum Covering by Cliques, and Maximum Independent Set of a Chordal Graph", *SIAM J. on Computing,* Vol. 1, pp. 180-187, 1972.

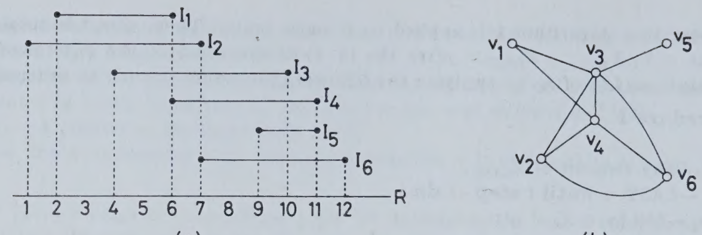
[6] F. Gavril, "Algorithms for a Maximum Clique and a Maximum Independent Set of a Circle Graph", *Networks,* Vol. 3, pp. 261-273, 1973.

[7] M.C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs,* Academic Press, New York, NY, 1980.

[8] W.L. Hsu, "Maximum Weight Clique Algorithms for Circular-arc Graphs and Circle Graphs", *SIAM J. on Computing,* Vol. 14, pp. 224-231, 1985.

[9] S. Masuda, K. Nakajima, T. Kashiwabara and T. Fujisawa, "Efficient Algorithms for Maximum Cliques of an Overlap Graphs", Technical Report, Systems Research Center, University of Maryland, College Park, MD, 1986.

[10] D. Rotem and J. Urrutia, "Finding Maximum Cliques in Circle Graphs", *Networks,* Vol. 11, pp. 269-278, 1981.


I1
I2
I3
I4
I5
I6
R
1 2 3 4 5 6 7 8 9 10 11 12
v1
v5
v3
v4
v2
v6

(a) (b)

Fig.1 (a) A family of intervals. (b) Its corresponding overlap graph.

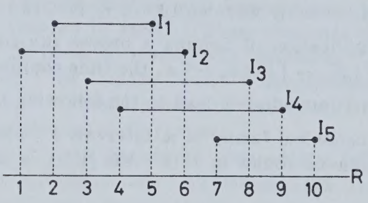
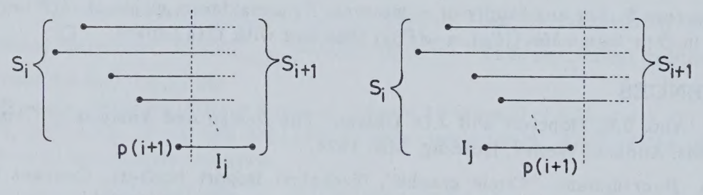

I1
I2
I3
I4
I5
R
1 2 3 4 5 6 7 8 9 10

Fig.2 A canonical family of intervals.


S_i
S_{i+1}
p(i+1)
I_j
S_i
S_{i+1}
I_j
p(i+1)

(a) (b)

Fig.3 The relationship between S_i and S_{i+1}.
(a) $p(i+1)$ is the left endpoint of I_j. (b) $p(i+1)$ is the right endpoint of I_j.

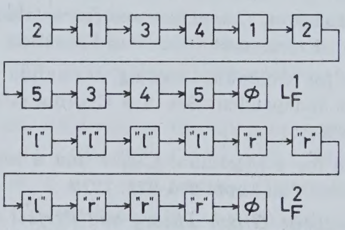

2 1 3 4 1 2
5 3 4 5 φ L_F^1
"l" "l" "l" "l" "r" "r"
"l" "r" "r" "r" φ L_F^2

Fig.4 The lists L_F^1 and L_F^2 representing the family of intervals of Fig.2.

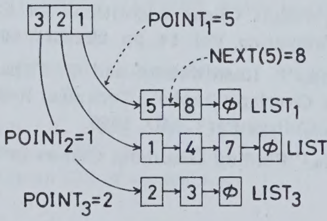

3 2 1
POINT1=5
NEXT(5)=8
5 8 φ LIST1
POINT2=1
1 4 7 φ LIST2
POINT3=2
2 3 φ LIST3

Fig.6 Data structures for λ-representation.

MAXIMUM ECONOMICAL DOMINATION IN A TREELIKE NETWORK

JAMES McHUGH
YEHOSHUA PERL

C.I.S. Department
N.J.I.T.
Newark, New Jersey 07102

ABSTRACT

We consider the problem of locating service centers in a treelike network in order to maximize the serviced population under budget constraints. We show that the problem is NP-Hard. In the case the costs of establishing the service centers are equal for all n cities we obtain the maximum weight k-domination problem. An $O(nk^2)$ dynamic programming procedure is given. Finally an $O(nB^2)$ pseudo polynomial dynamic programming procedure is presented for the original problem, where B is the budget constraint.

1. INTRODUCTION.

Let G(V,E) be a graph, where the vertices represent cities, and there is an edge between a pair of cities if the corresponding cities are mutually easily accessible. We assume there is a weight associated with each city which represents its population.

We consider the problem of locating service centers at the cities, under the assumption that each service center can serve only the home city itself and its adjacent cities. We wish to optimize the location of the service centers, so that the centers service the maximum possible population. We remark that if a city is serviced by more than one service center, its population is still only counted once.

The best possible outcome would be that every city was serviced by at least one service center, either in the city itself or in an adjacent city. However, it often occurs that a budget precludes attaining this maximum possibility. Thus, suppose there is a cost c(v) to set up a service center in city v, for every v in V. Suppose also there is a total budget B which can be spent establishing service centers. Then, we may look for a set S of cities with $\mathrm{COST}(S) = \sum_{v \in S} c(v) \leq B$ and such that the population which will be served by service centers in S will be maximized.

Supported in part by the State of New Jersey SBR Grants.

The relevant mathematical concept is that of Domination. A set of vertices S is said to dominate all the vertices which are adjacent to S. For our application, a vertex is considered as dominating not only its adjacent vertices, but also the vertex itself, corresponding to the fact that a service center serves not only its adjacent cities, but its home city as well. Thus, in our case we consider weighted domination where each dominating vertex is responsible for the weight of itself and the adjacent vertices. Thus, we are looking for a maximum weighted dominating set S under the budget constraint $\sum_{S} c(v) \leq B$. This is the constrained maximum weight domination problem.

The original domination problem was to find a minimum set of vertices which together dominate all the other vertices in a graph. This was shown to be NP-Hard by Karp[6]. Recently, much attention has been given to variations of the Domination Problem. See, for example, the extensive bibliography by Hedetniemi and Laskar[5].

It is clear that our extension of this problem is difficult. Actually, we shall show the problem is NP-Hard even for trees, for which case there is a linear algorithm for the original domination problem[1]. The minimum cost domination problem for weighted trees is solved in linear time by Natarajan and White[8]. In this problem the costs of the domination is the sum of costs of the dominating vertices and the cost of the least edges from all other vertices to a dominating vertex. We refer to [4] for a bibliography on domination on trees. In this work, we limit ourselves to trees.

First, we show a reduction from the knapsack problem [6] to our problem for a tree, thus showing our problem is NP-Hard. Then we consider the case where the cost of an installation of a service center at every vertex is the same. Thus, we have equal costs $c = c(v)$. In this case, the budget constraint is translated into building $k = \lfloor B/c \rfloor$ service centers. In this case, the problem is to find a set of k cities maximizing the sum of the populations of those k cities and their adjacent cities. Let D(S) denote the set of vertices in V-S which are adjacent to vertices in S. Then, our problem is to find a set S of k vertices such that $\mathrm{WEIGHT}(S) = \sum_{v \in S \cup D(S)} w(v)$ is maximized. Denote this problem as the maximum weight k-domination problem. We present an $O(nk^2)$ dynamic programming algorithm for the problem. The special case where $w(v) = 1$ for every vertex v in V was solved by Hsu[3]. Note the difference between our model and some other models of optimum location of the service centers in a network. For example in [7] and [2] the centers provide services to all vertices and the objective is to minimize the average service cost. In our model only part of the vertices , namely those assigned service centers and their adjacent vertices, are receiving service and the objective is to maximize the population served by the k centers. Finally an $O(nB^2)$ pseudo polynomial dynamic programming procedure is presented for the constrained maximum weight domination problem.

2. NP-COMPLETENESS RESULT.

In this section we show that the maximum weight domination problem, under the cost constraint $\sum_{S} c(v) \leq B$, where S is the dominating set, is NP-Hard even when the graph is a tree.

Let us first formulate the corresponding decision problem.

The constrained weight domination problem :

Given a tree T with n vertices having non-negative integral weight w_i and cost c_i associated with each vertex i, and two integers M and B, is there a dominating set S in T satisfying $COST(S) \leq B$ and $WEIGHT(S) \geq M$?

This problem is clearly in NP. The reduction is from the NP-complete knapsack problem (see Karp[6]):

Given a set U of elements $u_1, u_2, \ldots, u_n$ with integer weight $w(u_i)$ and cost $c(u_i)$ associated with each u_i, and two integers B and M, is there a subset U' of U such that:

$$\sum_{U'} w(u_i) \geq M \quad \text{and} \quad \sum_{U'} c(u_i) \leq B \ ?$$

The reduction is as follows : Construct a rooted tree T of 2n+1 vertices where the root r, with $w(r) = 0$ and $c(r) = B+1$, has n sons $v_1, \ldots, v_n$ with weights $w(v_i) = 0$ and cost c_i. Each son v_i has a single son v_i' with $w(v_i') = w_i$ and $c(v_i') = B+1$. Refer to Figure 1 for an illustration.

Clearly a dominating set of cost at most B uses only a subset U' of the vertices v_i, since all the other vertices violate the cost constraint. The weight of such a dominating set is just $\sum_{i \in U'} w_i$ and its cost is just $\sum_{i \in U'} c_i$. Thus, if we can solve the constrained weighted dominataion problem in polynomial time, then we could solve the knapsack problem in polynomial time. This establishes that the constrained weight domination problem is NP-Complete even for trees.

3. DYNAMIC PROGRAMMING PROCEDURE FOR THE MAXIMUM WEIGHT K-DOMINATION PROBLEM.

In the previous section we show that the constrained maximum weight domination problem is NP-Hard. In this section we consider the special case of equal costs which as we showed earlier is actually the maximum weight k-domination problem.

Note that transforming the tree into a rooted tree does not change the problem. Thus we may assume that a tree rooted at a vertex r is given.

We shall describe a dynamic programming procedure which is polynomial in n and k. The algorithm applies a bottom-up scanning technique to the (rooted) tree, calculating for each vertex u and integer k', $0 \leq k' \leq k$, the weight WD(u,k') of a maximum weight k'-dominating set in the subtree rooted at u. In case k' is greater than the size of the subtree at u, we define $WD(u,k) = -\infty$. The last computed value WD(r,k) is the weight of the maximum weight k-dominating set in the original tree.

Let the subtree T(u) be the complete subtree of T rooted at the vertex u, and let $x_1,\ldots,x_m$ be the m sons of u. Let $T(x_1),\ldots,T(x_m)$ be the complete subtrees rooted respectively at $x_1,\ldots,x_m$. Refer to Figure 2 for an illustration.

The development of the algorithm is complicated by two features. First, the analysis depends critically on whether the root u of T(u) does or does not lie in a maximum weight dominating set, as well as on whether the root is or is not dominated by a maximum weight dominating set. Secondly, the straight forward dynamic programming procedure is computationally intractable.

For example, in the case that the root of the subtree T(u) is neither in, nor dominated by, the dominating set, then the principle of optimality which gives the basis for the dynamic programming procedure can be applied in a straightforward manner. In this case, a maximum weight k-dominating set for T(u) is the disjoint union of maximum weight k_i-dominating sets for the subtrees $T(x_1),\ldots,T(x_m)$, where $\sum_{i=1}^{m} k_i = k$. But, even in this simplified case, if we follow a straight forward dynamic programming approach, then for each sequence $k_1,\ldots,k_m$ such that $\sum_{i=1}^{m} k_i = k$, the corresponding maximum weight k_i-dominating sets must be found. Since there are $O(k^{m-1})$ such sequences to be considered, this is intractable for large values of m.

We can avoid the computational difficulties of this direct approach by using the following propagation technique. We let $T_i'(u)$, $i \leq m$ denote the u-rooted subtree of T spanned by $T(x_1),\ldots,T(x_i)$. We then iteratively construct a maximum weight k'-dominating set, for each $k' \leq k$, for each of the subgraphs $T_i'(u)$, $1 \leq i \leq m$. For $i = m$, we obtain the desired maximum weight k'-dominating set for the tree T(u). This technique leads as we shall see, to a polynomial time algorithm. For another example of such a propogation technique see [9].

We address the complications caused by the possible states of u (in the dominating set or not, dominated by the dominating set or not) by defining subproblems appropriate to the status of u. We then solve these interrelated subproblem by a bottom-up, left-right dynamic programming scheme that iterates over all the subtrees $T_i'(u)$, for each vertex u in T.

We propose the following definitions :

WD(u,k',i,1) : the weight of the maximum weight k'-dominating set in the tree $T_i'(u)$, where u is required to lie in the dominating set.

WD(u,k',i,0) : the weight of the maximum weight k'-dominating set in the tree $T_i'(u)$, where u does not lie in the dominating set.

WDE(u,k',i,1) : the weight of the maximum weight k'-dominating set in $T_i'(u)$, where u is dominated by the dominating set, but does not lie in the dominating set.

WDE(u,k',i,0) : the weight of the maximum weight k'-dominating set in $T_i'(u)$, where u is neither in, nor dominated by the dominating set.

WD(u,k',1) : the weight of the maximum weight k'-dominating set in T(u), where u is in the dominating set.

WD(u,k',0) : the weight of the maximum weight k'-dominating set in T(u), where u is not in the dominating set.

WD(u,k') : the weight of the maximum weight k'-dominating set in T(u).

WD(r,k) is the quantity actually sought.

The initializations for i = 0 are as follows :

$$WD(u,k',o,1) = \begin{array}{ll} W(u) & \text{for } k'=1 \\ -\infty & \text{for } k'<>1, \end{array} \tag{1.1}$$

$$WD(u,k',0,0) = \begin{array}{ll} 0 & \text{for } k'=0 \\ -\infty & \text{for } k'>o \end{array} \tag{1.2}$$

$$WDE(u,k',o,1) = -\infty \quad \text{for all } k' \tag{1.3}$$

$$WDE(u,k',0,0) = \begin{array}{ll} 0 & \text{for } k'=0 \\ -\infty & \text{for } k'>0 \end{array} \tag{1.4}$$

These assignments also suffice to define the initial values for the endpoints of the tree.

The general dynamic programming relations between the variables and the proofs of these relations follow.

The first relation we establish is :

$$\begin{array}{rl} WD(u,k',i+1,1) = \max\limits_{0\le k''\le k'} & WD(u,k'',i,1)+WD(x_{i+1},k'-k'',1), \\ & WD(u,k'',i,1)+WDE(x_{i+1},k'-k'',1), \\ & WD(u,k'',i,1)+WDE(x_{i+1},k'-k'',0)+W(x_{i+1}), \end{array} \tag{2}$$

where x_{i+1} refers here the to i+1 son of u.

The principle of optimality applies here as follows. WD(u,k',i+1,1) gives the weight of the maximum weight k'-dominating set in $T_{i+1}'(u)$ that includes u. We can partition such a set into that part of it that lies in $T_i'(u)$ and that part of it that lies in the complete tree rooted at x_{i+1}, viz., $T(x_{i+1})$. The part of the dominating set in $T_i'(u)$ includes u by definition, and may be chosen to be the maximum weight such set. Its size can be any integer k" from 0 to k'. The remaining part of the dominating set lies in $T(x_{i+1})$ and has size k'-k". This part has to be handled differently depending on the status of x_{i+1}, viz., according as :

(i) x_{i+1} is in the dominating set, or

(ii) x_{i+1} is not in the dominating set but is dominated by it, or

(iii) x_{i+1} is neither in the dominating set, nor is dominated by it.

One must exercise care in calculating the weight of the combined parts of the dominating set. Case (i) is straight forward. The weight of the maximum part in $T(x_{i+1})$ must be $WD(x_{i+1},k'-k'',1)$. In this case, x_{i+1} is in the dominating set, so its weight is already accounted for, and hence the weight dominated by the combined cover is given by the first expression in the maximization in (2). Case (ii) is similar. However, in case (iii), the weight of x_{i+1} is not covered by that part of the dominating set lying in $T(x_{i+1})$. Since u is in the part of the dominating set in $T_i'(u)$, then x_{i+1} is dominated by the combined parts. Thus, its weight must be added on explicitly as in the third expression in (2).

The next relation is :

$$WD(u,k',i+1,0) \;=\; \max_{j=0,1} \; WDE(u,k',i+1,j). \qquad (3)$$

This is not an application of the principle of optimality. It merely converts an i+1-order WD problem into a pair of i+1-order WDE problems. The following relations solve those WDE problems.

$$\begin{aligned} WDE(u,k',i+1,1) = \max_{0\le k''\le k'} \; & WDE(u,k'',i,1)+WD(x_{i+1},k'-k'',1), \\ & WDE(u,k'',i,1)+WD(x_{i+1},k'-k'',0), \qquad (4) \\ & WDE(u,k'',i,0)+WD(x_{i+1},k'-k'',1)+W(u) \end{aligned}$$

Considerations similar to those for calculating WD(u,k',i+1,1) apply here. The vertex u is dominated by, but does not lie in the dominating set. There are three cases according as u is dominated :

(i) both by the part of the dominating set in $T_i'(u)$ and the part in $T(x_{i+1})$, or

(ii) by the part in $T_i'(u)$, but not the part in $T(x_{i+1})$, or

(iii) by the part in $T(x_{i+1})$, but not the part in $T_i'(u)$.

Case (i) corresponds to the first expression in (4). For case (ii), observe that requiring that u is not dominated by x_{i+1} implies x_{i+1} is not in the part of the dominating set lying in $T(x_{i+1})$. Therefore, the second expression in (4) uses the term $WD(x_{i+1},k'-k'',0)$, which excludes x_{i+1} from the part of the dominating set in $T(x_{i+1})$. Finally, for case (iii), u is excluded and x_{i+1} is included in the dominating set, whence the first two terms in the third expression in (4). The combined parts dominate u, giving the third term in this expression.

The final WDE relation is :

$$WDE(u,k',i+1,0) = \max_{0\le k''\le k'} \; WDE(u,k'',i,0) + WD(x_{i+1},k'-k'',0) \qquad (5)$$

This relation merely states that u must be both excluded from and not dominated by the dominating set in $T_i'(u)$, whence the term WDE(u,k",i,0), also, x_{i+1} cannot lie in the part of the dominating set in $T(x_{i+1})$, otherwise u would be dominated by it, whence the term $WD(x_{i+1},k'-k'',0)$, which excludes x_{i+1}.
The final relations to observe are :

$$WD(u,k',i) = WD(u,k',deg(u),i), \qquad (i=0,1) \tag{6}$$

$$WD(u,k') = \max_{i=0,1} WD(u,k',i). \tag{7}$$

4. COMPLEXITY ANALYSIS.

The calculation of WD(r,k) depends on a sequence of previously calculated WD-terms of the form : WD(u,k',j), WD(u,k',i,j), and WDE(u,k',i,j), where u varies over the N vertices of T, k' varies from 0 to k, i varies from 1 to degree(u), and j is either 0 or 1. The over all complexity of the calculation is $O(nk^2)$. To prove this, we argue as follows. Referring to (2), observe that calculating WD(u,k',i+1,1) entails O(k') operations on the previously calculated WD-terms. WD(u,k',1) is just WD(u,k',deg(u),1), so its calculation requires O(k'deg(u)) operations on previously found WD-terms. Since this must be done for all $0 \leq k' \leq k$, the work at each vertex is $O(k^2 deg(u))$. Summing over all the vertices gives

$$\sum_{\text{all } u} O(k^2 deg(u)) = O(nk^2) \text{ operations in all to}$$

calculate WD(r,k).

It is straightforwrd to maintain within the comlexity bound for every variable the appropriate list of dominating vertices. Thus the maximum weight k-dominating set is obtained together with WD(r,k).

The algorithm analysis for the maximum k-domination problem shows the dynamic programming procedure for WD(r,k) actually defines a pseudo-polynomial algorithm for the constrained maximum weight domination problem. The notation for the solution for the general problem is similar to that for the maximum k-domination solution. The term WD(u,B',i,1) now refers to the maximum weight dominating set in $T_i'(u)$, where u is included in the dominating set and the cost of the set is B'. The other terms are defined similarly. For the defining relations, we merely replace occurences of k in the given relations with occurences of a general integer cost B, and make some minor modifications at the initialization phase. WD(r,B) is the weight of the constrained maximum weight dominating set. The resulting algorithm has performance $O(nB^2)$.

5. CONCLUDING REMARKS.

In this paper, we have considered the cost constrained maximum weight tree domination problem. We have shown this problem to be NP-Hard. For the special case of equal costs, we have described an $O(nk^2)$ dynamic programming procedure for the problem of finding a maximum weight k-dominating set. A pseudo polynomial dynamic programming procedure is then presented for the general constrained maximum weight domination problem for trees.

REFERENCES

[1] Cockayne, E.J., Goodman, S.E., and Hedetniemi, S.T., A Linear Algorithm for the Domination Number of a Tree, Information Processing Letters, 4(1975), 41-44.

[2] Fischer, M.J., Guibas, L.J., Griffeth, N.D., Lynch, N.A., Optimal Placement of Identical Resources in a Distributed Network, Proc. 2nd Int. Conf. on Distributed Computing Systems, Paris, April, 1981, pp 324-336.

[3] Hsu, W-L. On the Domination Numbers of Trees, North-western University, Dept. Ind. Eng. & Management Sci., Tech. Rept., July, 1982.

[4] Hedetniemi, S., Hedeteniemi, S., and Laskar, R. Domination in Trees: Models and algorithms, in Graph Theory and its Applications to Algorithms and Computer Science, edit. by Y. Alavi, et al, John Wiley & Sons, Inc., 1985, pp 423-441.

[5] Hedetniemi, S. and Laskar, R. A Bibliography on Domination, Clemson Univ., Nov. 1985.

[6] Karp, R.M. Reducibility among Combinatorial Problems, in R.E.Miller and J.W. Thatcher (eds), Complexity of Computer Computations, Plenum Press, 1972, pp 85-104.

[7] Kariv, O. and Hakimi, S.L. , An Algorithmic Approach to Network Location Problems Part 1: The P-centers, SIAM J. APPL. MATH. 37, 1979, pp 513-538.

[8] Natarajan, K.S. and White, L.J. Optimum Domination in Weighted Trees, Information Precessing Letters, 7(1978), 261-265.

[9] Perl, Y. , and Shiloach, Y. , Efficient Optimization of Monotonic Functions on Trees, SIAM J. of Algebriac and Discrete Methods, 4(1983), pp 512-516.

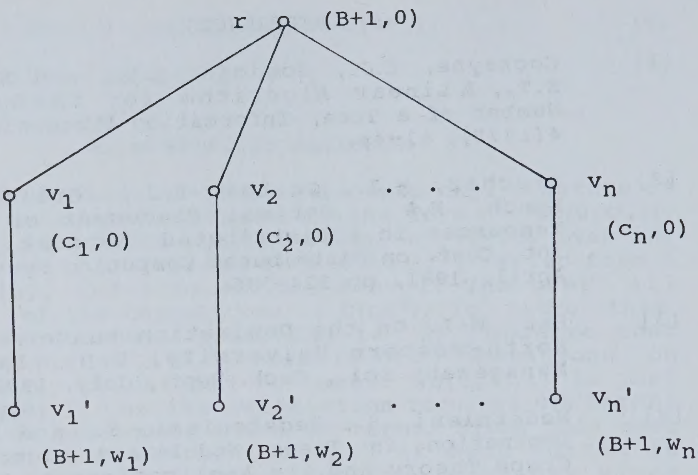

r
$(B+1,0)$
v_1
$(c_1,0)$
v_2
$(c_2,0)$
. . .
v_n
$(c_n,0)$
v_1'
$(B+1,w_1)$
v_2'
$(B+1,w_2)$
. . .
v_n'
$(B+1,w_n)$

Figure 1. Knapsack reduction

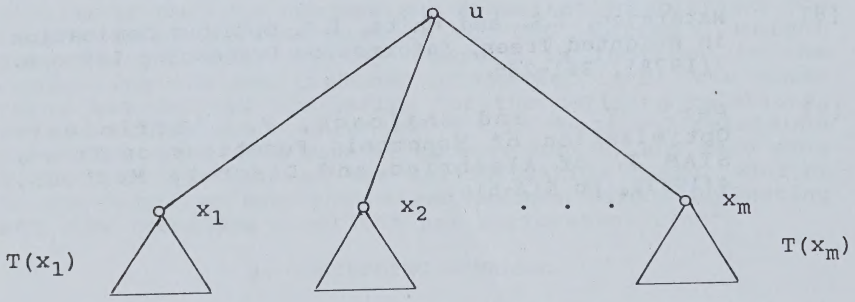

u
x_1
x_2
. . .
x_m
$T(x_1)$
$T(x_m)$

Figure 2. Tree T(u) Rooted at u.

STABILITY IN LINEAR ESTIMATION

P.A. KELLY
Dept. of Electrical
and Computer Engineering
Univ. of Massachusetts
Amherst, MA 01003

W.L. ROOT
Aerospace Engineering Dept.
The University of Michigan
Ann Arbor, MI 48109

ABSTRACT

This paper is concerned with the stability of linear estimation of the amplitude coefficients of deterministic signals in noise. A class of estimators having nominal performances constrained to be close to that of the nominal linear unbiased minimum-variance (LUMV) estimator is specified. A stability index is defined that is, for each estimator, a measure of the sensitivity of the estimate distribution to inaccuracy in the observation model. The estimator for which the stability index is minimized is then found.

I. INTRODUCTION

The topic of this paper is the stability of estimators of the coefficients of a linear combination of known signals in noise. The observation space is taken to be $Z = L_2[0,T]$. The nominal model for the observation $z \varepsilon Z$ is

$$z = \sum_{k=1}^{L} b_k s^{(k)} + w \tag{1}$$

where each signal $s^{(k)}$ is a known element of Z; the noise w is a Z-valued random variable having zero mean and covariance operator Λ; and $b_1, \ldots b_L$ are the unknown real numbers to be estimated. The null space of Λ, $\eta(\Lambda)$, is assumed to be zero. Let $\{\psi^{(k)}, k = 1,\ldots,L\}$ be the usual orthonormal basis for R^L. Let $b = \sum_{k=1}^{L} b_k \psi^{(k)}$, and let $S: R^L \to Z$ be the linear operator given by $Sb = \sum_{k=1}^{L} b_k s^{(k)}$; then (1) may be written as $z = Sb + w$. We assume that each $s^{(k)}$ is in $R(\Lambda)$ (the range of Λ) and that $\{s^{(k)}, k = 1,\ldots,L\}$ is a linearly independent set. Then S has rank L, and $S^*\Lambda^{-1}S: R^L \to R^L$ is well-defined and invertible (where "*" denotes adjoint).

We are interested in linear estimates of b; that is, estimates of the form $\hat{b} = Hz$ for some linear operator $H: Z \rightarrow R^L$. The estimator H is unbiased if and only if $HS = I_L$ (the identity operator on R^L). If H is unbiased, then the variance of the estimate $\hat{b}$ is equal to

$$E(||\hat{b} - b||^2) = E(||H\ w\ ||^2)$$
$$= \sum_{k=1}^{L} (H\ \Lambda\ H^* \psi^{(k)}, \psi^{(k)}) = \mathrm{Tr}\ [H\ \Lambda\ H^*]. \tag{2}$$

Theorem 2.2 of [1] can be applied (as is discussed in [5]) to show that, if the supposed model is correct, a linear unbiased minimum-variance (LUMV) estimator H_o exists as the continuous extension to all of Z of the densely defined operator

$$(S^*\ \Lambda^{-1} S)(\Lambda^{-1/2} S)^* \Lambda^{-1/2}.$$

The variance of the estimate $\hat{b}_o = H_o z$ is equal to

$$V_o \triangleq \mathrm{Tr}[H_o\ \Lambda\ H_o^*] = \mathrm{Tr}[(S^* \Lambda^{-1} S)^{-1}]. \tag{3}$$

If w is Gaussian noise, then $\hat{b}_o$ is also the maximum likelihood estimate of b.

The concern of this paper is with situations in which the observation does not exactly fit the model. For example, the true signal space may only be approximated as the span of $\{s^{(1)}, \ldots, s^{(L)}\}$, and Λ may only approximate the true noise covariance. Similar problems have been considered previously in the context of matched filtering. In [2], Poor found minimax-robust matched filters (i.e., filters maximizing worst-case signal-to-noise ratios) for problems of one signal in noise, with signal and noise covariance uncertainty. Chen and Kassam [3] derived robust matched filters for multiple-input systems. The problem treated here is different because the quantities of interest are parameter estimators rather than matched filters. The approach is also different. We define a class of linear estimators that perform to within specified constraints under nominal conditions, and then seek the most stable member of that class; i.e., the one for which some appropriately defined stability index is minimized. Despite these differences, the stable estimators that we find have, in many cases, forms reminiscent of matched filters for signal uncertainty.

II. OPTIMALITY CRITERION AND STABILITY INDEX

Suppose first that in the observation model (1), $L = 1$ and w is nominally Gaussian. This case was treated in [4] - [6], with emphasis on the related detection problem. The measure of nominal performance, called the optimality index, was chosen to be the unit-amplitude signal-to-noise ratio (which is the inverse of the variance for an unbiased amplitude estimate). Uncertainty was introduced in the model by allowing the actual probability measure on $B(Z)$, the Borel sets of Z, to vary in a Prokhorov neighborhood of the nominal. It was shown that the largest possible difference between the actual and nominal distributions of the estimate or detection statistic is then determined by a scalar quantity, called the stablilty index. The problem of minimizing this index subject to a constraint on the optimality index was then solved.

The intention here is to follow an approach similar to that taken in [4] - [6] for the more general estimation problem with $L \geq 1$. We restrict attention to nominally unbiased linear estimators. The criterion for optimality of such an estimator H is based on a comparison of how it performs under nominal conditions with how the LUMV estimator H_o performs. We thus require as an optimality constraint that the nominal variance of the estimate formed using H be acceptably close to the minimum variance V_o. That is, for some choice of $\delta > 0$, we require that

$$\mathrm{Tr}\,[H \Lambda H^*] \leq V_o + \delta. \tag{4}$$

Unfortunately, the choice of a stability index is not as well-motivated as in the one-signal case. To some extent, a Prokhorov neighborhood argument can be made to justify the scalar index we prefer to use. Let μ be the nominal probability measure on $B(Z)$ corresponding to the observation (1) with a fixed value of $\{b_1, \ldots, b_L\}$. Assume that the true measure ν lies in a Prokhorov neighborhood of μ. That is, ν is an element of

$$M_{\varepsilon,\eta}(\mu) \overset{\Delta}{=} \{\nu: \nu(A^{\varepsilon}) \leq \mu(A^{\varepsilon})+\eta \quad \text{for all } A \in B(Z)\} \tag{5}$$

where A^{ε} is the closed ε-neighborhood of A. (See [7], Chapter 2.) This is a quite general representation of model uncertainty. In particular, it includes the case in which the true observation is in the form $z = Sb + w + n$, where n is an extra independent noise governed by a measure ω such that

$$\omega \{z: ||z|| \leq \varepsilon\} \geq 1 - \eta.$$

Let F_{μ} and F_{ν} be the distributions of $\hat{b}$ under the measures μ and ν, respectively, and define $h^{(k)} = H^{*}\psi^{(k)}$. Then

$$F_{\nu}(t_1,\ldots,t_L) = \nu\{z: (z,h^{(1)}) < t_1,\ldots, (z,h^{(L)}) < t_L\}$$

$$\leq \mu\{z: (z,h^{(1)}) < t_1,\ldots, (z,h^{(L)}) < t_L\}^{\varepsilon}+ \eta$$

$$< \mu\{z: (z, h^{(1)}) < t_1 + \varepsilon||h_1||,\ldots, (z, h^{(L)}) < t_L + \varepsilon||h_L||\} + \eta$$

$$= F_{\mu}(t_1 + \varepsilon\, ||h_1||,\ldots, t_L + \varepsilon||h_L||) + \eta.$$

Now, $\nu \in M_{\varepsilon,\eta}(\mu)$ implies that $\mu \in M_{\varepsilon,\eta}(\nu)$. Hence, the above argument can be repeated with the roles of μ and ν reversed to show that

$$F_{\nu}(t_1,\ldots,t_L) \geq F_{\mu}(t_1- \varepsilon\, ||h_1||,\ldots,t_L- \varepsilon\, ||h_L||)- \eta.$$

That is , the actual distribution of $\hat{b}$ is bounded by functions that are, except for the added η, translated versions of the nominal distribution, with the squared norm of the translation equal to

$$\varepsilon^2 \left[\sum_{k=1}^{L} ||h^{(k)}||^2\right] = \varepsilon^2\, \mathrm{Tr}[HH^{*}].$$

Now, the nominal variance of $\hat{b}$ also depends on H, and the larger the variance of $\hat{b}$, the less effect a translation has on the shape of its distribution. We thus choose as a normalized stability index the squared norm of the translation divided by the variance; that is,

$$(\mathrm{Tr}[H \Lambda H^*])^{-1} \mathrm{Tr}[HH^*]. \tag{6}$$

This is a measure of the sensitivity of the distribution of $\hat{b}$ to inaccuracy in the model. If $L = 1$, it is the stability index used in [4] - [6].

III. STABLE LINEAR ESTIMATORS

Our goal is to minimize the stability index over the class of unbiased estimators meeting the optimality constraint. Hence, we have the problem:

(I) Given $\delta > 0$, find the linear operator H minimizing $(\mathrm{Tr}[H \Lambda H^*])^{-1}$ $\mathrm{Tr}[HH^*]$ subject to $HS = I_L$ and $\mathrm{Tr}[H \Lambda H^*] \leq V_o + \delta$.

To solve this, we consider two related problems:

(II) Given $\delta > 0$, find H minimzing $\mathrm{Tr}[HH^*]$ subject to $HS = I_L$ and $\mathrm{Tr}[H \Lambda H^*] = V_o + \delta$; and

(III) Given $\eta > -1$, find H minimizing $\mathrm{Tr}[HH^*] + \eta\, \mathrm{Tr}[H \Lambda H^*]$ subject to $HS = I_L$.

Let I be the identity operator on Z. For $\eta > -1$, define

$$H(\eta) = [S^*(I + \eta\Lambda)^{-1}S]^{-1} S^*(I + \eta\Lambda)^{-1} \tag{7}$$

and

$$V(\eta) = \mathrm{Tr}[H(\eta) \Lambda H^*(\eta)]. \tag{8}$$

Without loss of generality, we assume that $||\Lambda|| = 1$. The following preliminary result is proved in [5]:

Lemma 1. For any $\eta > -1$, $H(\eta)$ is the unique solution to problem (III). Furthermore, if for a given $\delta > 0$ there is some $\eta > -1$ such that $V(\eta) = V_o + \delta$, then $H(\eta)$ is the unique solution to problem (II).

The remaining results in this paper do not appear in [5]. In order to use Lemma 1 to find a solution to problem (I), some additional facts are needed. Proofs are outlined.

Lemma 2. $\lim_{\eta\to\infty} V(\eta) = V_o$.

Proof. Expand eq. (8) for $V(\eta)$ by substituting in the value of $H(\eta)$ from eq. (7). It can be shown that

$$[S^*(I+\eta\Lambda)^{-1}S]^{-1} = \eta[(S^*\Lambda^{-1}S)^{-1} + A(\eta)]$$

and $$[S^*(I+\eta\Lambda)^{-1}\Lambda(I+\eta\Lambda)^{-1}S] = \frac{1}{\eta^2}[(S^*\Lambda^{-1}S) + B(\eta)]$$

where $||A(\eta)|| \to 0$ and $||B(\eta)|| \to 0$ as $\eta \to \infty$. Hence,

$$V(\eta) \to \mathrm{Tr}[(S^*\Lambda^{-1}S)^{-1}] = V_o \text{ as } \eta \to \infty.$$

Lemma 3. $V(\eta)$ is a monotone decreasing function of η for $\eta > -1$. If it is true that:

(A) $R(S) \neq R(\Lambda(I+\eta\Lambda)^{-1}S)$ for any $\eta > -1$

then $V(\eta)$ is strictly monotone decreasing.

Proof. The first part of the lemma follows from calculating $-\frac{d}{d\eta}V(\eta)$ and noting that, for $\eta > -1$, it is equal to the trace of a non-negative definite $L\times L$ matrix $Q(\eta)$. Condition (A) is sufficient to imply that $Q(\eta) \neq 0$ for any $\eta > -1$, so $\frac{d}{d\eta}V(\eta) = -\mathrm{Tr}[Q(\eta)] < 0$.

Let $f(\eta)$ denote the value of the stability index (6) evaluated at $H = H(\eta)$.

Lemma 4. Suppose that

(B) $R(S) \not\supset R((I-f(\eta)\Lambda)(I+\eta\Lambda)^{-1}S)$ for any $\eta > -1$.

Then no $H(\eta)$ for $\eta > -1$ locally minimizes the stablity index on the set $\{H: HS = I_L\}$.

Proof. Suppose that $H(\eta)$ locally minimizes the stability index on $\{H: HS = I_L\}$. Then Thm. 1 on p. 243 of [8] may be used to show that it must be true that

$$H(\eta)\ (I-f(\eta)\Lambda) = CS^*$$

for some L×L matrix C. By substituting in eq.(7) for H(η), this can be seen to be equivalent to

$$(I-f(\eta)\Lambda)\ (I+\eta\Lambda)^{-1}S = SD$$

for some L×L matrix D. Condition (B) implies that no such matrix exists.

Now, by Lemma 3 and the continuity of V(η), $V_1 \overset{\Delta}{=} \lim_{\eta\downarrow -1} V(\eta)$ exists. Assume that condition (A) in Lemma 3 holds. Then $V_1 > V_o$. Define $\delta_1 = V_1 - V_o$. Then Lemmas 2 and 3 imply that for each $0 < \delta' < \delta_1$ there is a unique $\eta' > -1$ such that $V(\eta') = V_o + \delta'$; in fact, η' decreases monotonically as δ' increases. Then by Lemma 1, H(η') is the unique solution to problem (II) with $\delta = \delta'$. On the other hand, if condition (B) in Lemma 4 holds, then no H(η) locally minimizes the stability index on $\{H{:}HS = I_L\}$. We can thus state the following:

Theorem 1. Assume that conditions (A) and (B) hold. Let δ be given, with $0 < \delta < \delta_1$. Then there is an $\eta > -1$ such that H(η) is the unique solution to problem (I), with $V(\eta) = V_o + \delta$. The value of η decreases monotonically as δ increases.

Proof. Let η' be such that H(η') is the unique solution to problem (II) with δ=δ', and let H' be a solution to problem (I) for the same value of δ. Then it can be shown that H' must equal H(η'') for some $\eta'' \geq \eta'$. If $\eta'' > \eta'$, then $V(\eta'') < V(\eta') = V_o + \delta'$. Then H(η'') locally minimizes the stabilty index on the set $\{H{:}\ HS = I_L,\ \mathrm{Tr}[H\ \Lambda\ H^*] \leq V_o+\delta'\}$. But this implies that H(η'') locally minimizes the stability index on $\{H{:}\ HS = I_L\}$, which cannot be true. Thus, η'' = η' and H' = H(η').

So, subject to conditions (A) and (B), problem (I) can be solved for a range of values of δ. We note that these conditions would usually be satisfied in practice. In fact, let $\{\phi_1, \phi_2, ...\}$ be orthonormal eigenfunctions of Λ in some order, and let K be the sum of the L largest multiplicities of eigenvalues of Λ. Define $\underline{s}_m \varepsilon R^L$ by

$\underline{s}_m = [(s^{(1)}, \phi_m), \ldots, (s^{(L)}, \phi_m)]^*$. It can then be shown that both conditions (A) and (B) are implied by:

(C) There are more than K values of m for which $\underline{s}_m$ is non-zero.

Suppose that δ is such that $V_o + \delta < V(0) = \mathrm{Tr}[(S^*S)^{-1}(S^*\Lambda S)(S^*S)^{-1}]$. Then $H(\eta)$ for some $\eta > 0$ solves problem (I). Note that $H(\eta)$ can be written as

$$H(\eta) = [S^*(\eta^{-1}I+\Lambda)^{-1}S]\ S^*\ (\eta^{-1}I+\Lambda)^{-1}.$$

That is, $H(\eta)$ differs from H_o in that the operator corresponding to the noise covariance is changed from the nominal value Λ to $\Lambda+\eta^{-1}I$. This is similar to the way the robust matched filters for signal uncertainty found in [2] vary from the nominally optimal filters.

In one special case we can state an explicit expression for the value of V_1 and solve problem (I) for $\delta \geq \delta_1$. Let $P: Z \to Z$ be the orthogonal projection onto the eigenspace of Λ corresponding to its largest eigenvalue. Then the following can be proved in a manner similar to the proof of Lemma 2:

Lemma 5. If S^*PS is non-singular, then $V_1 = \mathrm{Tr}[(S^*PS)^{-1}]$.

This result can then be used in solving problem (I) for $\delta \geq \delta_1$.

Theorem 2. If S^*PS is non-singular, then for $\delta \geq \delta_1$ a solution to problem (I) is given by $H_1 = (S^*PS)^{-1}S^*P$.

Proof. Note that $\mathrm{Tr}[H_1H_1^*] = V_1 = V_o + \delta_1$. Also, if λ is the largest eigenvalue of Λ, then $\mathrm{Tr}[H_1\ \Lambda\ H_1^*] = \lambda\ \mathrm{Tr}[H_1H_1^*]$. Since for any H, $\mathrm{Tr}[H\ \Lambda\ H^*] \leq \lambda\ \mathrm{Tr}[HH^*]$, H_1 gives the minimum possible value of the stability index.

ACKNOWLEDGEMENT

This research was supported in part by the Office of Naval Research under Grant N00014-85-K-0149.

REFERENCES

[1] F.J. Beutler and W.L. Root, "The operator pseudo inverse in control and systems identification," in Generalized Inverses and Applications, M.Z. Nashed, Ed. New York: Academic Press, 1976.

[2] H.V. Poor, "Robust matched filters", IEEE Trans. Inform.Theory, vol. IT-29, pp. 677-687, 1983.

[3] C.T. Chen and S.A. Kassam, "Robust multiple-input matched filtering: frequency and time-domain results", IEEE Trans. Inform.Theory, Vol. IT-31, pp. 812-821, 1985.

[4] P.A. Kelly and W.L. Root, "Stable Linear Detectors with Optimality Constraints", Proc. 22nd Allerton Conf. on Commun., Contr., and Comp., 1984.

[5 ———, Stability of detection of signals in noise," Proc. 23rd IEEE Conf. Decis. and Contr., 1984.

[6] ———, Stability in linear detection", to appear, IEEE Trans. Inform. Theory, 1987.

[7] P.J. Huber, Robust Statistics. New York: Wiley, 1981.

[8] D.G. Luenberger, Optimization by Vector Space Methods. New York: Wiley, 1969.

Lℓ - FILTERS

FRANCESCO PALMIERI, CHARLES G. BONCELET Jr.
Department of Electrical Engineering
University of Delaware, Newark, DE 19716

SUMMARY

A growing interest recently has been devoted to the definition of nonlinear filters that can combine optimality and robustness. It is well known that the performance of optimal filters can degrade spectacularly in the presence of occasional outliers in the data or with seemingly minor deviations of the model from the real world.

The recent theory of robust estimation has formulated the problem of finding estimates which are insensitive to minor variations. Most available results refer to the estimation of a location parameter or to cases in which only a small number of parameters need to be extracted from the data (Huber [1]). Martin and Thomson [2] reported ways of dealing with spectral estimation in a robust way; Masreliez and Martin [3], and Boncelet and Dickinson [4], proposed ways of robustifying the Kalman filter. Kassam and Poor [5], among others, have dealt with the determination of minimax solutions for detection in which families of distributions or spectra are assumed.

Our intention in this paper is to define a new type of data smoother within the class of nonlinear estimators based on ordered observations. The starting point for our work is the contribution by Lloyd [6] that showed how the location parameter can be more efficiently estimated as a linear combination of ordered samples than by the classical sample mean. The corresponding mean square error will always be smaller than or equal to that obtained with the sample mean or the sample median. In situations with heavy tailed noises, the mean square error can be much smaller.

Peterson and Kassam [10] suggested a generalized version of the order statistic filter that incorporates time information, but they did not propose any computational methods for determining the coefficients.

Bovik et al. [7] described a class of "order statistic" or L-filters. These filters are a simplified form of Lloyd's estimate applied to the problem of filtering a noisy signal. In a previous work [8], we have applied these ideas to the problem of filtering a noisy, nonconstant signal. A filter based on a linear combination of the ranked samples (L-filter) can be designed for such a case to minimize the mean square error. In [8], we showed how the coefficients of the L-filter can be readily computed for the model of a Markov sequence as the signal corrupted by an additive, independent noise process.

L-filters are in general not robust; however they can be replaced by robust alternatives in the same way that the sample mean is replaced by an alpha trimmed mean: arbitrarily trim off, or eliminate, some of the largest and some of the lowest samples. The resulting estimator will be resistant to a number of outliers.

In some cases in which the corrupting noise has heavy tailed density functions, the L-filter can perform better than the best linear transversal filter (FIR). However, the most important disadvantage of the L-filter is that time information is lost in the ordering process. Of course, for a constant signal embedded in i.i.d. noise, time ordering is irrelevant; however, for nonconstant signals in independent noise this is not so. (This result is obvious for Gaussian signals corrupted by a Gaussian noise process.) In the paper, we present numerical examples for Markov signals in non-Gaussian noise which illustrate that the L-filter performance peaks at a finite, non-zero window size and deteriorates as the window is widened.

Accordingly, we have generalized the class of L-filters by incorporating time information in the filter structure. We call this class "Lℓ-filters." The class includes both

linear filters and order statistic filters. There are two equivalent ways of representing these filters: 1. An Lℓ-filter is an L-filter with the coefficient of the i^{th} ordered sample dependent on the *time* position of the i^{th} ordered sample, or 2. the Lℓ-filter is a linear FIR filter with each coefficient dependent on the *ordered* position of particular sample. These filters are more difficult to design than L-filters—in particular, an n^2 by n^2 matrix must be inverted, but not much harder to implement.

The breakdown probability as defined by Mallows [9] can be used to choose the trimming coefficient that will determine the robustness of the smoother against occasional outliers and the corresponding coefficients can be obtained from an appropriate constrained minimization. The "linear part" of the filter [9], that represents the closest linear approximation to the filter for gaussian input, is used to explain the essential spectral behavior of the filter. Basically, we show that the essential frequency response of this nonlinear filter is not influenced by the ranking operation, i.e., the spectral response of the Lℓ-filter is essentially the same as that of the corresponding linear filter.

We propose a method of computing the coefficients which minimize the mean square error under various criteria, robustness, unbiasedness, location invariance etc. Numerical examples for the restoration of a Markov sequence corrupted by additive non-gaussian noise are shown which compare the performance of the Lℓ-filter with the classical median, mean, best L-estimates of location and the L-filter.

To lessen the design difficulties, we also suggest a simplified form that requires only iterated inversions of $n \times n$ positive definite matrices. This solution is basically the Kronecker product of an n-dimensional linear filter and an n-dimensional L-filter.

REFERENCES

[1] P. J. Huber, *Robust statistics,* John Wiley & Sons, 1981.

[2] R. D. Martin, D. J. Thomson, "Robust-resistant spectrum estimation", *Proc. of the IEEE,* vol. 70, no. 9, pp. 1097-1115, Sept. 1982.

[3] C. J. Masreliez, R. D. Martin, "Robust Bayesian estimation for linear models and robustifying the Kalman filter", *IEEE Trans. on Auto. Contr.,* vol. AC-22, no. 3, pp. 361-371, June 1977.

[4] C. G. Boncelet Jr., B. W. Dickinson, "An approach to robust Kalman filtering ", *Proc. 22nd IEEE Conference on Decision and Control (San Antonio, TX),* pp. 304-305, 1983.

[5] S. A. Kassam, H. V. Poor,"Robust techniques for signal processing: A survey", *Proc. of the IEEE,* vol. 73, no. 3, pp. 433-481, March 1985.

[6] E. H. Lloyd, "Least square estimation of location and scale parameters using order statistics", *Biometrika,* vol. 39, pp. 88-95, 1952.

[7] A. C. Bovik, T. S. Huang, D. C. Munson, "A generalization of median filtering using linear combinations of order statistics," *IEEE Trans. on Acoust., Speech, Signal Proc.,* vol. ASSP-31, pp. 1342-1350, Dec. 1983.

[8] F. Palmieri, C. G. Boncelet, "Optimal MSE combination of order statistics for restoration of Markov processes", *Proceedings of the Twentieth Annual Conference on Information Sciences and Systems (Princeton, NJ),* 1986.

[9] C. L. Mallows, "Some theory of nonlinear smoothers", *The Annals of Statistics,* vol. 8, no. 4, pp. 695-715, 1980.

[10] S. L. Peterson, S. A. Kassam, "General Order Statistic-Based Structures for Nonlinear Signal Restoration", *Proceedings of the Twentieth Annual Conference on Information Sciences and Systems (Princeton, NJ),* 1986.

A PARAMETRIC APPROACH FOR IMPULSE RESPONSE ESTIMATION

UMESH K. BHARGAVA
MIT Lincoln Laboratory
Lexington, MA 02173-0073

D. M. GOODMAN
Lawrence Livermore Nat. Lab.
Livermore, CA 94550

R. L. KASHYAP
Purdue University
W. Lafayette, IN 47907

INTRODUCTION

We consider the impulse response estimation of a linear time-invariant discrete system. Our approach is to assume that the transfer function of the system can be adequately modeled with a set of finite order numerator and denominator polynomials, and use a pair of input-output observations, made on the system, to estimate the coefficients of the polynomials. Let the coefficients of these polynomials be given by θ_i and β_i for $i = 1, 2 \cdots M$, respectively, where we assume that the order of the polynomials is known and is M; and let the input sequence, known precisely, be u(t), and the observed output sequence y(t) be the sum of the actual output and the additive white noise w(t) for $t = 1, 2, \cdots N$. We can thus write

$$Y(t) = \theta_1^0\, y(t-1) + \theta_2^0\, y(t-2) + \ldots \theta_M^0\, y(t-M) \qquad (1)$$

$$+\, w(t) + \theta_1^0\, w(t-1) + \ldots \theta_M^0\, w(t-M)$$

$$+\, \beta_1^0\, u(t-1) + \ldots \beta_M^0\, u(t-M),\ t = 1, 2, \ldots N.$$

The usual method for estimating θ_i and β_i is based on the assumption that w(t) has a Gaussian distribution; subsequently, an approximate maximum likelihood estimate (MLE) is found by minimizing an appropriate least squares criterion. The problem with such an approach is that the variance of the estimate is very large even if the actual distribution of w(t) is only slightly different from the Gaussian distribution; e.g., a distribu-tion from the family of mixture distributions given by a sum of zero mean Gaussian distributions one having a much larger variance than the other as in Eq. (2).

$$\beta(w) = (1-\varepsilon)\ \text{Gauss}\ (0, \sigma^2) + \varepsilon\ \text{Gauss}\ (0, a\sigma^2), \qquad (2)$$

Where $a >> 1$ and ε, the mixing parameter, is a positive number much smaller than 1.

Research has been supported by the David Ross grant from Purdue University, the office of the Naval Research under contract N00014-85K-0611, and the U.S. Dept. of Energy under contract W-7405-Eng-48.

A ROBUST METHOD

Our method is based on the minimax approach such that the maximum variance of our estimates of θ_i and β_i over all the distributions in the family of Eq. (2) is minimum for all possible estimates of θ_i and β_i. Using a saddle point property it can be shown that this estimate is obtained as in Eq. (3).

$$\theta = \text{argument} \min_{\theta} \sum \mathcal{H}(w(t, \theta)),$$

where

$$\mathcal{H}(x) = \begin{cases} x^2 & \text{if } |x| \leq c \\ c|x| - c^2/2 & \text{if } |x| > c, \end{cases} \tag{3}$$

$\theta = \text{col.}\,[\theta_1, \theta_2, \ldots \theta_M, \beta_1, \beta_2, \ldots \beta_M]$, c is an arbitrary constant between 1 and 2,

$$w(t, \theta) = y(t) - \theta_1 y(t-1) - \ldots \theta_M y(t-M) - \beta_1 u(t-1) - \ldots - \beta_M u(t-M) - \theta_1 w(t-1, \theta) - \ldots \theta_M w(t-M, \theta), \tag{4}$$

$w(t, \theta)$ is obtained recursively with $w(0, \theta) = 0$

By simulation we compared the sample mean squared error (MSE) in impulse response estimates from our approach with the one pertaining to the MLE under Gaussian assumption (Fig. 1). The input-output data sequences were taken from reference [1]. Figure 1 clearly shows that our estimates are more efficient compared to the MLE under gaussian assumption for the range $\cdot 01 \leq \varepsilon \leq \cdot 22$.

REFERENCES

[1] Bhargava, U.K., Kashyap R.L., Goodman, D.M., "Impulse response identification with deterministic inputs using nonparametric methods," Proc. of 23rd Annual Allerton Conference on Comm, Control and Computing, Oct. 1985, pp 584-593.

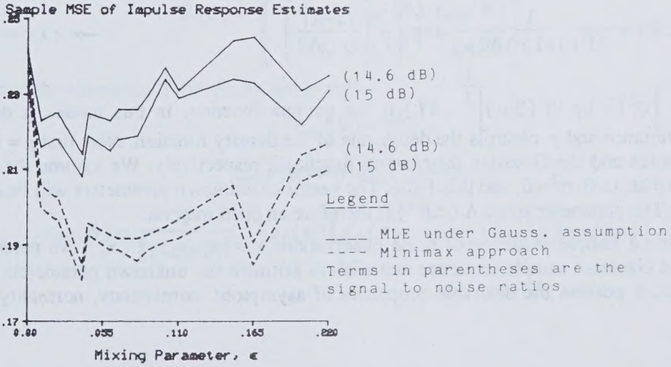


Fig. 1

PARAMETER ESTIMATION FOR THE GENERALIZED GAUSSIAN NOISE MODEL

Mahesh K. Varanasi and Behnaam Aazhang

Electrical and Computer Engineering Dept.
Rice University
P.O. Box 1892
Houston, TX 77251

1. INTRODUCTION

In many applications, modern communication systems must operate in complex and variable noise environments that are often dominated by non-Gaussian man-made and natural sources. The structured character of these interfering sources can significantly degrade the performance of conventional systems, which are designed to operate most effectively against the commonly assumed Gaussian background noise processes [1]. By proper system design this noise structure can often be exploited to yield optimal schemes for better performance than would be obtained in the Gaussian case. A classic example is that of locally optimum detection of coherent signals in the presence of additive white non-Gaussian noise [8]. These detectors consist of a zero-memory nonlinearity followed by a matched filter whose output is compared with a threshold. The nonlinearity depends entirely on the noise probability density function. Results such as these can therefore be implemented if we can properly characterize the noise environment.

In this study, we assume that the noise samples are independent, identically distributed (i.i.d) with distribution belonging to, or closely approximated by the generalized Gaussian family. Members of this parametric family of distributions are used to model the first order characteristics of a number of noise phenomena. For instance, fixed kernal density estimates for non-Gaussian processes in the ocean acoustic environment have been shown to agree closely with generalized Gaussian densities with different exponential decay rates [6]. In this paper, we wish to present asymptotically efficient and computationally expedient procedures for estimating the parameters of the generalized Gaussian density which characterizes the channel noise.

The generalized Gaussian noise model

The channel observations are sampled, yielding the stochastic sequence $\mathbf{X} = (X_1, X_2, \cdots, X_n)$. We parametrically characterize the noise distribution by modelling this sequence to be i.i.d with the distribution corresponding to the generalized Gaussian density function. This parametric family of densities is obtained by generalizing the Gaussian density to obtain a variable rate of exponential decay and is given by

$$f(x;\boldsymbol{\theta}) = \frac{1}{2\Gamma(1+1/p)A(p)} \exp\left\{-\left[\frac{|x-\mu|}{A(p)}\right]^{p}\right\} \qquad -\infty < x < \infty \qquad (1.1)$$

where $A(p) = \left[\sigma^2 \Gamma(1/p)/\Gamma(3/p)\right]^{\frac{1}{2}}$, $\Gamma(.)$ is the gamma function. In this model, μ denotes the mean, σ^2, the variance and p controls the decay rate of the density function. Note that $p=1$ and $p=2$ yield the Laplacian and the Gaussian distribution functions, respectively. We assume the parameter values are such that $p>0$, $\sigma^2>0$, and μ is finite. The vector of unknown parameters will be denoted as $\boldsymbol{\theta} = [\mu, \sigma^2, p]$. The parameter space Λ ($\subset R^3$) is therefore an open interval.

Thus, given a sample sequence of noise observations $\mathbf{x} = (x_1, x_2, \cdots x_n)$, we model it as an i.i.d generalized Gaussian sample sequence and wish to estimate the unknown parameters, requiring that our estimators possess the desirable properties of asymptotic consistency, normality and high relative efficiency. To this effect, we use three methods which are most suitable for the estimation of the parameters of a generalized Gaussian density function. These methods have been successfully implemented and we point out in this paper their relative merits and demerits through an analysis of the variance of the estimators and the computational complexity involved in each of the three methods.

As the term *asymptotic* suggests, we will, in each of the three methods that follow, consider **x** not for a fixed n, but as a member of a sequence corresponding to $n = N, N+1 \cdots$. Therefore, any estimate $\theta_n = \underline{f}(\mathbf{x})$ is really a *sequence of estimates*. We will denote the corresponding *estimator sequence* as $\boldsymbol{\Theta}_n = \mathbf{f}(\mathbf{X})$. In what follows, we will simply refer to θ_n and $\boldsymbol{\Theta}_n$ as an *estimate* and an *estimator*, respectively, instead of the more accurate but cumbersome terms, *sequence of estimates* and *estimator sequence*. Asymptotic properties of estimators are therefore properties of the sequence of random vectors $\boldsymbol{\Theta}_n$ as $n \longrightarrow \infty$.

2. THE MOMENT METHOD

The moment method is a simple, intuitively appealing and computationally expedient estimation technique. It consists of equating an appropriate number of the sample moments to the corresponding moments of the distribution which are functions of the unknown parameters. By considering as many moments as there are parameters, estimates of the latter are obtained. The method of moments estimator possesses the desirable asymptotic properties of $\sqrt{n}$-consistency and normality [3]. It is in general *not* asymptotically efficient in the sense that the asymptotic variances of the estimators do not achieve the Cramer-Rao lower bound.

Let us denote the moment method estimate as $\bar{\theta}_n = [\bar{\mu}_n, \bar{\sigma}_n^2, \bar{p}_n]$ and the corresponding estimator as $\bar{\boldsymbol{\Theta}}_n = [\bar{M}_n, \bar{V}_n, \bar{P}_n]$. Therefore $\bar{\mu}_n$ and $\bar{\sigma}_n^2$ are simply the sample mean and the sample variance respectively whereas an iterative root finding technique is used to solve for $\bar{p}_n$ from the r^{th} sample central moment

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{\mu}_n)^r = \frac{[A(\bar{p}_n)]^r \Gamma((r+1)/\bar{p}_n)}{\Gamma(1/\bar{p}_n)}, \tag{2.1}$$

with the constraint that $\bar{p}_n > 0$. r is an even integer ≥ 4. As was noted before, $\bar{\boldsymbol{\Theta}}_n$ is asymptotically normally distributed so that

$$\sqrt{n}\,(\bar{\boldsymbol{\Theta}}_n - \theta) \xrightarrow{i.d.} \eta\left[0; \Sigma(\theta)\right] \tag{2.2}$$

where $\eta(\mu; \mathbf{C})$ denotes the multivariate Gaussian probability density function with mean vector μ and covariance matrix **C**. The asymptotic performance of the moment method estimator is therefore completely specified by $\Sigma(\theta)$, which in our problem for $r=4$, has the form

$$\Sigma(\theta) = \begin{bmatrix} \varepsilon_{\mu\mu}(\theta) & 0 & 0 \\ 0 & \varepsilon_{\sigma^2\sigma^2}(\theta) & \varepsilon_{\sigma^2 p}(\theta) \\ 0 & \varepsilon_{\sigma^2 p}(\theta) & \varepsilon_{pp}(\theta) \end{bmatrix}, \tag{2.3}$$

where each of the elements of the matrix are given as

$$\varepsilon_{\mu\mu}(\theta) = \sigma^2, \tag{2.4}$$

$$\varepsilon_{\sigma^2\sigma^2}(\theta) = \sigma^4[g_4(p) - 1], \tag{2.5}$$

$$\varepsilon_{pp}(\theta) = \frac{1}{[g'_4(p)]^2}\left[4 g_4(p)[g'_4(p) - g_6(p)] + g_8(p) - g_4^2(p)\right], \tag{2.6}$$

and

$$\varepsilon_{\sigma^2 p}(\theta) = \frac{\sigma^2}{g'_4(p)} \left[g_4(p) - 2 g_4^2(p) + g_6(p) \right] . \qquad (2.7)$$

The function $g_r(p)$ depends only on r and p and is defined as

$$g_r(p) \overset{\Delta}{=} \frac{1}{\sigma^r} \frac{[A(p)]^r \Gamma(r+1/p)}{\Gamma(1/p)}$$

and $g'_4(p)$ is the derivative of $g_4(p)$ with respect to p.

Clearly, from (2.3), the MM estimator of μ is asymptotically independent of the estimators of σ^2 and p, while the latter are correlated. Furthermore, the asymptotic variance of the estimator of σ^2 decreases sharply as p increases for $p<1$. Further, the asymptotic variance of the estimator of p does not depend on σ^2.

Maximum likelihood estimation of $\boldsymbol{\theta}$ will be considered next, followed by a comparative study of the moment method and the asymptotically efficient maximum likelihood estimation method.

3. MAXIMUM LIKELIHOOD ESTIMATION (single parameter case)

In many applications, one often encounters the problem of estimating one of the parameters when the others are known. We will present in this section, the estimation of each of the three parameters of the generalized Gaussian density function assuming the other two are given.

The likelihood function of the sample **x** having independent components is defined to be

$$\mathrm{L}(\mathbf{x};\theta) = \log \prod_{i=1}^{n} f(x_i;\theta).$$

THEOREM 1

Under mild regularity conditions, there exists [5] *some* solution $\hat{\theta}_n$ of the likelihood equation

$$\frac{\partial \mathrm{L}(\mathbf{x};\theta)}{\partial \theta} = 0 ,$$

such that the corresponding estimator $\hat{\Theta}_n$ is asymptotically consistent if $f'(x;\theta)$ exists for almost all x and for all $\theta \in \Lambda$. Further, this consistent solution is asymptotically normal and efficient [3] and is unique and corresponds to the relative maximum of the likelihood function [4], provided the following conditions are satisfied.

(R1) For almost all x,

$$\frac{\partial \log f(x;\theta)}{\partial \theta} , \frac{\partial^2 \log f(x;\theta)}{\partial^2 \theta} , \frac{\partial^3 \log f(x;\theta)}{\partial^3 \theta} \text{ exist } \forall\, \theta \in \Lambda,$$

(R2) Let θ_0 be the true value of θ. Then $\forall\, \theta \in \Lambda$,

$$\left| \frac{\partial f(x,\theta)}{\partial \theta} \right| < F_1(x) , \left| \frac{\partial^2 f(x,\theta)}{\partial \theta^2} \right| < F_2(x) , \left| \frac{\partial^3 \log f(x,\theta)}{\partial \theta^3} \right| \le H(x) ,$$

where F_1, F_2 are integrable over $(-\infty, +\infty)$ and $E_{\theta_0}[H(x)] < \infty$, $\forall x$

(R3) $\forall\, \theta \in \Lambda$, $0 < I(\theta) < \infty$ where $I(\theta)$ is Fisher's Information and is defined as

$$I(\theta) = E_\theta \left\{ \left[\frac{\partial \log f(x;\theta)}{\partial \theta} \right]^2 \right\}.$$

Thus if the likelihood equation (LE) has a unique root, and Λ is an open interval, any sequence of solutions $\hat{\theta}_n$ of the LE corresponds to the maximum likelihood estimate (MLE) of θ and is asymptotically normal and efficient. i.e.,

$$\sqrt{n}\,(\hat{\theta}_n-\theta) \xrightarrow{i.d.} \eta\left[0;[I(\theta)]^{-1}\right]$$

The applicability of Theorem 1 to the problem of estimating each of the parameters of the generalized Gaussian density function will now be investigated. Furthermore, an extension of Theorem 1 is stated, the proof of which can be found in [10], which enables us to show the efficiency of the ML estimator of μ in some region of the parameter space Λ where (R1)-(R3) do not hold. In addition, an exact distribution for the ML estimator of σ^2 is obtained.

Estimation of mean given variance and p

The first derivative of the likelihood function

$$\frac{\partial}{\partial\mu}\log f(x;\theta)=p(A(p))^{-p}\,sgn(x-\mu)\mid x-\mu\mid^{p-1}$$

does not satisfy (R1) for all (or almost all) x for $p \le 1$ since $\forall x, \exists$ a value $\mu = x$ at which it does not exist. No claim about the consistency of the solution of the LE can therefore be made for $p<1$. However, it can be shown that conditions (R1)-(R3) hold for $p=2$ and for $p>3$. Further, since the generalized Gaussian density function is strictly, strongly unimodal, and differentiable for $p>1$ the LE has a unique solution in probability.

Using an extension of Theorem 1, the proof of which can be found in [10] , the following proposition holds. Note that it is not a direct consequence of Theorem 1.

Proposition 1 : The maximum likelihood estimate $\hat{\mu}_n$ obtained as the solution of the LE

$$\frac{\partial L(\mathbf{x};\theta)}{\partial\mu}=\frac{p}{[A(p)]^p}\sum_{i=1}^{n}\operatorname{sgn}(x_i-\hat{\mu}_n)\mid x_i-\hat{\mu}_n\mid^{p-1}=0\,, \tag{3.1}$$

is such that the estimator $\hat{M}_n$ is asymptotically consistent, normal and efficient for all $p\ge 2$.

If $\hat{M}_n$ is efficient, its asymptotic variance is given by

$$asy.var(\hat{M}_n)=\frac{1}{nI_{\mu\mu}(\theta)}=\left[n\frac{p^2}{\Gamma(1/p)A^2(p)}\Gamma\left(\frac{2p-1}{p}\right)\right]^{-1}. \tag{3.2}$$

For $p=1$, the solution of the LE (3.1) yields $median(X_i)$ for $\hat{M}_n$ which is known to be asymptotically efficient. Therefore, $\hat{M}_n$ is asymptotically normal and efficient for $p=1$ and for $p\ge 2$. Further, it may be reasonable to expect efficiency of $\hat{M}_n$ for all $p\ge 1$.

Estimation of variance given mean and p

Conditions (R1)-(R3) hold. Further, the likelihood equation

$$\frac{\partial L(\mathbf{x};\theta)}{\partial\sigma^2}=\frac{p}{2\hat{\sigma}_n^2}\sum_{i=1}^{n}\left[\frac{|x_i-\mu|}{A(p)}\right]^p-\frac{n}{2\hat{\sigma}_n^2}=0 \tag{3.3}$$

has a unique real positive solution, $\hat{\sigma}_n^2$, the MLE of σ^2, and can be obtained as

$$\hat{\sigma}_n^2=\left[\frac{p}{n}\sum_{i=1}^{n}\left[\frac{|x_i-\mu|}{c(p)}\right]^p\right]^{2/p}, \tag{3.4}$$

where $c(p)=[\Gamma(1/p)/\Gamma(3/p)]^{1/2}$. Thus $\hat{V}_n$, the corresponding estimator is asymptotically consistent, normal and efficient for all $\theta\in\Lambda$. Furthermore, the asymptotic variance of $\hat{V}_n$ is given by

$$asy.var(\hat{V}_n) = \frac{1}{n I_{\sigma^2\sigma^2}(\theta)} = \left[nE \left(\frac{\partial \log f(x)}{\partial(\sigma^2)} \right)^2 \right]^{-1} = \frac{4\sigma^4}{np}. \tag{3.5}$$

Consider now a stronger result. An exact distribution for $\hat{V}_n$ can be obtained.

Proposition 2 : The maximum likelihood estimator of the variance of the generalized Gaussian density function given mean and p had an exact probability density function given as

$$f_{\hat{V}_n}(v) = \frac{p}{2c\Gamma(n/p)2^{n/p}} \left[\frac{v}{c} \right]^{n/2-1} \exp\left\{ -\frac{1}{2} \left[\frac{v}{c} \right]^{p/2} \right\} \quad 0 \le v < \infty \tag{3.6}$$

where $c = \left(\frac{p\sigma^p}{2n}\right)^{2/p}$.

Furthermore, the mean and the variance of $\hat{V}_n$ are given as

$$E[\hat{V}_n] = \sigma^2 (p/n)^{2/p} \frac{\Gamma((n+2)/p)}{\Gamma(n/p)} \tag{3.7}$$

$$var[\hat{V}_n] = \left[\frac{p\sigma^p}{n} \right]^{4/p} \left[\frac{\Gamma((n+4)/p)}{\Gamma(n/p)} - \left[\frac{\Gamma((n+2)/p)}{\Gamma(n/p)} \right]^2 \right] \tag{3.8}$$

Note from equation (3.7) that $\hat{V}_n$ is unbiased for $p=2$, and has a positive bias σ^2/n for $p=1$.

Proposition 3 : For large n, the second order approximations for the mean and the variance of $\hat{V}_n$, are given as

$$E[\hat{V}_n] \sim \sigma^2 \left[1 + \frac{(2/p-1)}{n} \right] \tag{3.9}$$

$$var[\hat{V}_n] \sim \frac{4\sigma^4}{np} + \frac{\sigma^4}{n^2} \left[\left[\frac{4}{p} - 1 \right]^2 \left[2 - \frac{2p}{3} \right] - \left[\frac{2}{p} - 1 \right]^2 (13 - 8p) \right] \tag{3.10}$$

It follows from equations (3.9) and (3.10) that $\hat{V}_n$ is asymptotically unbiased and first order efficient, a fact pointed out earlier in the discussion.

Proofs of propositions 2 and 3 can be found in [10].

Estimation of p given mean and variance

Conditions (R1)-(R3) hold. Further, the likelihood equation

$$\frac{\partial L(x;\theta)}{\partial p} = 0 \tag{3.11}$$

has a unique root at the true value of p in probability [10]. Therefore any sequence of solutions, $\hat{p}_n$ of the LE (3.11) corresponds to the maximum likelihood estimator $\hat{P}_n$ which is asymptotically consistent, normal and efficient. Further, the asymptotic variance of $\hat{P}_n$ is given as

$$asy.var[\hat{P}_n] = \frac{1}{n I_{pp}(\theta)} = \left[nE \left[\frac{\partial \log f(x;\theta)}{\partial p} \right]^2 \right]^{-1} \tag{3.12}$$

where an expression for $I_{pp}(\theta)$ can be found in [10].

The asymptotic variance of $\hat{P}_n$ does not depend on σ^2. Recall that the same observation was made about the moment method estimator $\bar{P}_n$.

Remark : The solution of the LE (3.11) involves computing the digamma function $\psi(.)$. For an "arbitrarily fast" computation of the digamma function, see [10]. Considerable saving in computation resulted from the use of this fast procedure in solving the nonlinear equation.

4. MAXIMUM LIKELIHOOD ESTIMATION (multiparameter case)

In this section, we will consider the simultaneous estimation of multiple parameters.

THEOREM 2

Under conditions which are generalizations of (R1)-(R3) to the multiparameter estimation problem (referred to in the sequel by (M1)-(M3)), any sequence of solutions, $\hat{\theta}_n = [\hat{\theta}_{1n}, \cdots, \hat{\theta}_{kn}]$ of the likelihood equations (LE's)

$$\frac{\partial L(\mathbf{x};\theta)}{\partial \theta_j} = 0 \qquad j = 1, 2, \cdots, s$$

is such that the corresponding estimator $\hat{\boldsymbol{\Theta}}_n$ is asymptotically consistent, normal and efficient, provided the LE's have a unique root. Moreover, $\hat{\boldsymbol{\Theta}}_n$ is the maximum likelihood estimator provided a maximum likelihood estimate exists [5]. (Conditions under which the maximum likelihood estimate exists and is unique can be found in [7].) Therefore,

$$\sqrt{n}\,(\hat{\boldsymbol{\Theta}}_n - \boldsymbol{\theta}) \xrightarrow{i.d.} \eta\left[0; [I(\boldsymbol{\theta})]^{-1}\right] \tag{4.1}$$

where $I(\boldsymbol{\theta})$ is the Fisher's information matrix defined such that

$$I_{jk}(\boldsymbol{\theta}) = E_{\theta}\left[\frac{\partial \log f(x,\boldsymbol{\theta})}{\partial \theta_j}\,\frac{\partial \log f(x,\boldsymbol{\theta})}{\partial \theta_k}\right] \qquad j, k = 1, 2, \cdots, s$$

Simultaneous estimation of mean, variance and p

The conditions (M1)-(M3) hold for $p=2$ and $p \geq 3$. It can however be shown that for all $p \geq 2$, $\hat{\theta}_n = [\hat{\mu}_n, \hat{\sigma}_n^2, \hat{p}_n]$, the solution of the likelihood equations (3.1), (3.3) and (3.11) is such the corresponding estimator $\hat{\boldsymbol{\Theta}}_n = [\hat{M}_n, \hat{V}_n, \hat{P}_n]$ is asymptotically efficient. Further, we will assume that $\hat{\boldsymbol{\Theta}}_n$ is efficient for all $p \geq 1$ as we did for the estimation of mean. The Fisher's information matrix has the form

$$I(\boldsymbol{\theta}) = \begin{bmatrix} I_{\mu\mu} & 0 & 0 \\ 0 & I_{\sigma^2\sigma^2} & I_{\sigma^2 p} \\ 0 & I_{p\sigma^2} & I_{pp} \end{bmatrix}, \tag{4.2}$$

where $I_{\mu\mu}$, $I_{\sigma^2\sigma^2}$, I_{pp} are given by equations (3.2), (3.6) and (3.12). Further,

$$I_{p\sigma^2} = \frac{1}{2p^2}\left[1 + \psi(1+1/p) + \frac{1}{2}\left\{\psi(1/p) - 3\psi(3/p)\right\}\right]. \tag{4.3}$$

The ML estimator of mean is asymptotically independent of the estimators of variance and p, while the latter are correlated. Furthermore, the asymptotic variance of the estimator of p does not depend on σ^2. Similar observations were made about the MM estimator. We would like to add that the solution of the likelihood equations (3.1), (3.3) and (3.11) should be attempted only with the constraint $p > 1$ due to the non-existence of the gradient of the likelihood function at $x = \mu$ when $p \leq 1$.

Median as an estimate of mean when p ≤1

When $p \leq 1$, it is not difficult to show that the median $\bar{\mu}_n$ as an estimate of μ has an asymptotically normal distribution [10] such that

$$\sqrt{n}\,(\bar{M}_n-\mu)\xrightarrow{i.d.}\eta\left[0,\frac{\sigma^2\Gamma^3(1/p)}{p^2\Gamma(3/p)}\right] \tag{4.4}$$

The asymptotic variance of $\bar{M}_n$ has a maximum at $p=1$ and decreases rapidly as p decreases.

A practical solution for the three parameter problem for $p<1$ therefore consists of estimating the mean as the sample median $\bar{\mu}_n$ and then solving the likelihood equations (3.3) and (3.11). In fact, the problem of estimating σ^2 and p when μ is known is dealt with in detail in [10] where it is shown that (M1)-(M3) hold and also that the LE's have a unique root in probability.

5. COMPARISON OF THE MM AND MLE ESTIMATORS

Let $\hat{\Theta}_n$ and $\bar{\Theta}_n$ be the ML and MM estimator sequences of the parameter vector θ of the generalized Gaussian density function. In the region of $\bar{\Lambda}$ where (2.2) and (4.1) hold, the *Asymptotic Relative Efficiency* (ARE) of $\bar{\Theta}_n$ with respect to $\hat{\Theta}_n$ is defined to be the ratio of the sample sizes $\hat{n}/\bar{n}$ at which $\hat{\Theta}_{\hat{n}}$ and $\bar{\Theta}_{\bar{n}}$ perform "equivalently" (i.e., have confidence ellipsoids whose volumes are asymptotically equivalent in probability) and is given as [9]

$$\gamma_{MM}\overset{\Delta}{=}\left[\frac{|\,I(\theta)^{-1}\,|}{|\,\Sigma(\theta)\,|}\right]^{1/k} \tag{5.1}$$

where $|A|$ denotes the determinant of A and k denotes the number of parameters.
Plots of asymptotic relative efficiency of the single parameter MM estimator of each of the parameters μ, σ^2 and p and γ_{MM} for the 2-parameter problem of estimating σ^2 and p given μ, are shown in figures (i)-(iv). Also, γ_{MM} for the 3-parameter problem is shown in figure (v). Clearly, the MM estimators are quite inefficient for very small values of p corresponding to impulsive noise phenomena.

The asymptotic relative efficiencies of all MM estimators do not depend on μ or σ^2. Further, the moment method is highly efficient ($\gamma_{MM}>0.85$) when $1.8<p<4.9$ for the 2-parameter problem and for $1.7<p<4.1$ for the 3-parameter problem.

We conclude this section by noting that despite using the fast method for computing the digamma function, the 3-parameter problem was found to be computationally tedious as compared to its MM counterpart. This is especially true when a poor initial guess is chosen.

6. MOMENT/NEWTON-STEP ESTIMATION

This method combines the computational alacrity of the moment method and the asymptotic efficiency of the maximum likelihood estimator. It requires that some sequence of $\sqrt{n}$–consistent (but not necessarily efficient) estimators be available. Consider the following theorem.

THEOREM 3
Under conditions (M1)-(M3), if $\bar{\Theta}_n$ is a $\sqrt{n}$–consistent estimator of θ, the moment/Newton-step (MNS) estimate θ_n^*, obtained as the solution of the system of linear equations

$$n\sum_{k=1}^{s}(\theta_{kn}^*-\bar{\theta}_{kn})I_{jk}(\bar{\theta}_n)=L'_j(\bar{\theta}_n) \qquad j,k=1,2,\cdots,s \tag{6.1}$$

is such that the MNS estimator Θ_n^* is asymptotically efficient.

Since the MM estimator is $\sqrt{n}$–consistent and also highly efficient for the generalized Gaussian problem in most regions of practical interest of the parameter space, we will choose the MM estimator to be our starting sequence $\bar{\Theta}_n$. Note that solving the system of linear equations (6.1) requires computation of the gradient of the likelihood function and the information matrix only once. Thus, considerable saving in computation results in the single parameter efficient estimation of mean or variance since explicit expressions for first order efficient MNS estimates are obtained as

$$\mu_n^* = \bar{\mu}_n + \frac{\Gamma(1/p)}{np\,(A(p))^{p-2}\,\Gamma(2-1/p)} \sum_{i=1}^{n} sgn(x_i - \bar{\mu}_n)\,|\,x_i - \bar{\mu}_n\,|^{p-1} \qquad \forall\, p > 1 \tag{6.2}$$

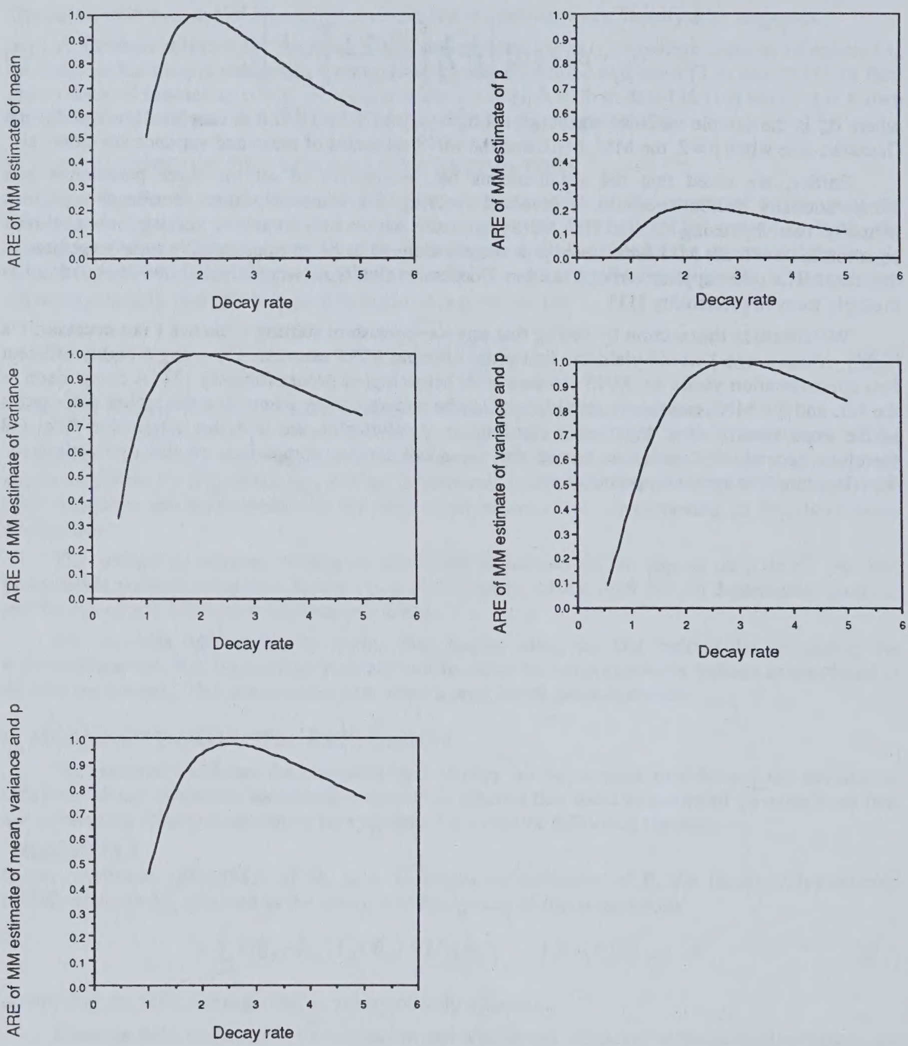
$$\sigma_n^{*2} = \bar{\sigma}_n^2 + 2\,\bar{\sigma}_n^2 \left[\frac{1}{n} \sum_{i=1}^{n} \left(\frac{|\,x_i - \mu\,|}{\bar{A}(p)} \right)^p - \frac{1}{p} \right] \tag{6.3}$$

where $\bar{\sigma}_n^2$ is the sample variance and $\bar{A}(p) = [\,\bar{\sigma}_n^2\,\Gamma(1/p)/\Gamma(3/p)\,]^{1/2}$. It is easy to show that for the Gaussian case when $p=2$, the MM, MLE and the MNS estimates of mean and variance are identical.

Earlier, we noted that the simultaneous ML estimation of all the three parameters was computationally tedious because it involved solving simultaneously three nonlinear equations (actually two, by using (3.4)). The MNS estimator which only involves solving one nonlinear equation to obtain the MM estimate of p is therefore found to be computationally more expedient in this case. The other application of Theorem 3 occurs in situations where the likelihood equation has multiple roots in probability [11].

We conclude this section by noting that any $\sqrt{n}$–consistent starting sequence (not necessarily a highly efficient one) would yield the first order efficient MNS estimate. Choosing a highly efficient first approximation yields an MNS estimate with better higher order efficiency [2]. A comparison of the ML and the MNS estimators should therefore be undertaken by comparing the higher order terms in the expansion of their asymptotic distribution. Analytically, we find this intractable. We will therefore depend on simulation results for the small-sample comparison of the two estimators. Simulation results are under preparation.

ASYMPTOTIC RELATIVE EFFICIENCY OF MM ESTIMATES



REFERENCES

[1] B. Aazhang, "Performance of DS/SSMA communications in impulsive channels," PhD Dissertation, Department of Electrical and Computer Engineering, University of Illinois, Urbana-Champaign, IL, 1986.

[2] M. Akahira and K. Takeuchi, *Asymptotic Efficiency of Statistical Estimators*. Springer, New York, 1981.

[3] H. Cramer, *Mathematical Methods of Statistics*. Princeton, 1946.

[4] V. S Huzurbazar, "The likelihood equation, consistency, and the maxima of the likelihood function," *Annals of Eugenics*, Vol.14, pp. 185-200, 1948.

[5] E.L Lehmann, *Theory of Point Estimation*. New York: John Wiley & Sons, 1983.

[6] F.W Machell and C.S Penrod, "Probability density functions of ocean acoustic noise processes," Tech. Rept. ARL-TP-82-37, Applied Research Lab, University of Texas, Austin,TX, 1982.

[7] T. Makelainen, K. Schmidt, and G. Styan, "On the existence and the uniqueness of the maximum likelihood estimate of a vector-valued parameter in fixed size samples," *Annals of Stat.*,, Vol.9, pp. 758-767, 1981.

[8] J.H Miller and J.B Thomas, "Detectors for discrete-time signals in non-Gaussian noise," *IEEE Trans. Comm.*,, Vol.COM-24, pp. 559-563, 1976.

[9] R. J Serfling, *Approximation Theorems of Mathematical Statistics*. Wiley, New York, 1980.

[10] M. K Varanasi, "Parameter Estimation of the Generalized Gaussian Noise Model," MS Thesis, Department of Electrical and Computer Engineering, Rice University, Houston, Texas, 1986.

[11] S.M Zabin, "Parameter estimation for the Class A Middleton model," Tech. Rept. R-1048, UILU-ENG 85-2223, Coordinated Science Lab., University of Illinois, Urbana-Champaign, IL, 1985.

MVSE ESTIMATION WITH AN MSE CONSTRAINT

JERRY D. GIBSON

Department of Electrical Engineering
Texas A&M University
College Station, TX 77843

ABSTRACT

A new estimation criterion is investigated which minimizes the variance of the squared error subject to a constraint on the mean squared error (MSE). Several theorems are given which characterize the estimator and its properties. Examples are presented that contrast the performance of the new estimator with the familiar minimum MSE estimator and the unconstrained minimum variance of the squared error estimator.

I. INTRODUCTION

Minimum mean squared error (MMSE) estimation has a long, illustrious history, and MMSE estimators are a critical component of many communications and control systems. However, estimators which minimize the mean squared error (MSE) ignore the variation of the MSE about its minimum value. In many applications, it is not only desirable to have a small MSE, but it is also necessary that the estimator MSE not be allowed to fluctuate too wildly, so that performance does not vary too much from the acceptable MSE. This paper is concerned with a new estimation criterion which minimizes the variance of the squared error subject to a constraint on the MSE. Actually, we treat a slightly more general problem by constraining the error variance rather than just the MSE. This new approach is called minimum variance of the constrained mean squared error (MVCMSE) estimation.

Sain and his students briefly investigated unconstrained minimum variance of the squared error (MVSE) estimation about twenty years ago [1–3]. Their work developed some theoretical results concerning MVSE estimation and demonstrated that the variance of the squared error could be reduced if the MSE is allowed to increase. They did not limit the increase of the MSE, however. Here we pose and solve the MVSE problem subject to a constraint on the MSE, and as we shall see, including the constraint not only can produce different estimators but it can also aid in solving some otherwise difficult examples. The motivating applications for the present problem are those where an estimator serves as the input to another adaptive system, such as an adaptive controller or adaptive signal processor. The concept for this work came from Sain's research in stochastic control on minimizing the variance of a performance index subject to a constraint on the mean of the performance index [4].

II. PROBLEM FORMULATION

Similar to [1], the problem to be solved can be stated as follows. Given a signal process $s(t)$, the process $g(t)$ is defined as

$$g(t) = T[s(t)], \tag{1}$$

where $T[\cdot]$ is some possibly nonlinear transformation. We desire an estimate $\hat{g}(t)$ of $g(t)$ which is a linear functional of some observed process, $\{x(t),\, t \in I\}$, where I is the index set. Thus,

$$\hat{g}(t) = L_g[x(\tau)],\ \tau \in I, \tag{2}$$

where I is the set of admissible time points and L_g is a linear operator. The estimate $\hat{g}(t)$ is to be chosen to minimize the criterion

$$\begin{aligned} J_{MVC} = E\{[(g(t) - L_g[x(\tau)])^2 - E\{(g(t) - L_g[x(\tau)])^2\}]^2\} \\ + \lambda E\{[(g(t) - L_g[x(\tau)]) - E(g(t) - L_g[x(\tau)])]^2 - C\} \end{aligned} \tag{3}$$

where λ is selected to yield

$$E\{[(g(t) - L_g[x(\tau)]) - E(g(t) - L_g[x(\tau)])]^2\} = C, \tag{4}$$

the acceptable value of the error variance. Of course, C is greater than or equal to the minimum error variance, or for $E\{g(t) - L_g[x(\tau)]\} = 0$, $C \geq MMSE$. The solution to the above–stated problem is obtained in the following section, and it is called the MVCMSE estimator, and denoted by the operator L^*_{MVC}.

In order to evaluate the utility of the MVCMSE estimator, it is important to compare its performance to that of the familiar linear MMSE estimator and the linear MVSE estimator [1–3]. We henceforth drop the modifier "linear" in the sequel.

The MMSE estimator for this same problem minimizes the performance index

$$J_{MM} = E\{[g(t) - L_g[x(\tau)]]^2\} \tag{5}$$

and satisfies the orthogonality principle given by

$$E\{[g(t) - L^*_M[x(\tau)]]x(\gamma)\} = 0, \tag{6}$$

for all $\gamma \in I$, where L^*_M denotes the MMSE estimator. The MVSE estimator is shown in [1–3] to satisfy the "projection principle"

$$E\{[(g(t) - L^*_{MV}[x(\tau)])^2 - E\{(g(t) - L^*_{MV}[x(\tau)])^2\}](g(t) - L^*_{MV}[x(\tau)])x(\gamma)\} = 0 \tag{7}$$

for all $\gamma \in I$, where L^*_{MV} denotes the MVSE estimator. For the sequel, it is important to note that the solution of Eq. (7) will equal the solution of Eq. (6) if and only if [1]

$$E\{(g(t) - L^*_{MV}[x(\tau)])^3 x(\gamma)\} = 0 \tag{8}$$

for all $\gamma \in I$. Numerous other properties of the MVSE estimator are provided in [1].

III. THE MVCMSE ESTIMATOR

We now turn to the solution of the constrained MVSE problem specified in Eqs. (3) and (4) and derive several theorems which illustrate the properties of the MVCMSE estimator and aid us in finding this estimator for several specific examples. For ease of comparison, we parallel the development in [1] as much as possible.

Theorem I: The functional L^*_{MVC} defined by Eq. (2) and minimizing Eq. (3) subject to Eq. (4), if it exists, satisfies*

$$\begin{aligned} E\{[(g - L^*_{MVC})^2 - E\{(g - L^*_{MVC})^2\}](g - L^*_{MVC})x(\gamma) \\ + \frac{\lambda}{2}[(g - L^*_{MVC}) - E(g - L^*_{MVC})]x(\gamma)\} = 0, \end{aligned} \tag{9}$$

---

* For simplicity of notation, we suppress the arguments of the operators, functions, and random processes whenever possible.

for all $\gamma \in I$.

Proof: Define a new functional,

$$L' = L_{MVC}^{*} + \varepsilon L \tag{10}$$

for some small number ε and where L is an arbitrary linear functional on $x(\tau)$, and consider

$$\begin{aligned} J' = {} & E\{[(g - L_{MVC}^{*} - \varepsilon L)^2 - E\{(g - L_{MVC}^{*} - \varepsilon L)^2\}]^2\} \\ & + \lambda E\{[(g - L_{MVC}^{*} - \varepsilon L) - E(g - L_{MVC}^{*} - \varepsilon L)]^2\}. \end{aligned} \tag{11}$$

Then, a necessary condition for L_{MVC}^{*} to be the minimizing functional is that

$$\left.\frac{\partial J'}{\partial \varepsilon}\right|_{\varepsilon=0} = 0 \tag{12}$$

for arbitrary L. Letting $L[x(\gamma)] = x(\gamma)$ for each $\gamma \in I$ yields Eq. (9).•

To solve a problem using Theorem I, we first solve Eq. (9) for L_{MVC}^{*} in terms of λ, and then substitute this L_{MVC}^{*} into Eq. (4) for the chosen value of C to solve for λ.

If we define

$$C_{MVC} \triangleq E\{(g - L_{MVC}^{*})^2\} - E^2(g - L_{MVC}^{*}), \tag{13}$$

we can write Eq. (9) as

$$\begin{aligned} E\{[2(g - L_{MVC}^{*})^3 + (g - L_{MVC}^{*})[\lambda - 2C_{MVC} \\ -2E^2(g - L_{MVC}^{*})] - \lambda E(g - L_{MVC}^{*})]L\} = 0. \end{aligned} \tag{14}$$

Theorem II: The variance of the squared error for every L_{MVC}^{*} satisfying Eq. (9) can be found from

$$\begin{aligned} \text{var}\,\{(g - L_{MVC}^{*})^2\} = {} & E\{[(g - L_{MVC}^{*})^2 - E(g - L_{MVC}^{*})^2](g - L_{MVC}^{*})g\} \\ & + \frac{\lambda}{2} E\{[(g - L_{MVC}^{*}) - E(g - L_{MVC}^{*})]L_{MVC}^{*}\}. \end{aligned} \tag{15}$$

Proof: Expand the variance of the squared error as

$$\begin{aligned} \text{var}\,\{(g - L_{MVC}^{*})^2\} = {} & E\{[(g - L_{MVC}^{*})^2 - E\{(g - L_{MVC}^{*})^2\}] \cdot \\ & [(g - L_{MVC}^{*})(g - L_{MVC}^{*}) - E\{(g - L_{MVC}^{*})^2\}]\} \end{aligned} \tag{16}$$

and the result follows by simplification and using Eq. (9).•

Theorem III: If L_{MVC}^{*} satisfies Eq. (9) and produces a relative minimum of the variance of $(g - L_{MVC}^{*})^2$, then for any arbitrary L,

$$\begin{aligned} 6E\{L^2(g - L_{MVC}^{*})^2\} - 2E(L^2)E(g - L_{MVC}^{*})^2 \\ -4E^2\{L(g - L_{MVC}^{*})\} + \lambda[E(L^2) - E^2(L)] > 0. \end{aligned} \tag{17}$$

Proof: Follows directly from the requirement that the second variation be positive, i.e.,

$$\left.\frac{\partial^2}{\partial \varepsilon^2} J\right|_{\varepsilon=0} > 0.• \tag{18}$$

Corollary 1: If L^*_{MVC} satisfies Eq. (9) and produces a relative minimum of var$\{(g - L^*_{MVC})^2\}$, then

$$\begin{aligned} 6E\{(g - L^*_{MVC})^2 x^2(\gamma)\} - 2E\{x^2(\gamma)\}E\{(g - L^*_{MVC})^2\} \\ -4E^2\{(g - L^*_{MVC})x(\gamma)\} + \lambda[E(x^2(\gamma)) - E^2(x(\gamma))] > 0 \end{aligned} \tag{19}$$

Proof: Follows from Eq. (17) by letting $L = x(\gamma)$, $\gamma \in I$.•

Corollary 2: A sufficient condition for Eq. (17) to be satisfied is that

$$2E\{L^2(g - L^*_{MVC})^2\} - 2E(L^2)E\{(g - L^*_{MVC})^2\} + \lambda[E(L^2) - E^2(L)] \geq 0 \tag{20}$$

for every arbitrary L.

Proof: Since

$$E\{L^2(g - L^*_{MVC})^2\} - E^2\{L(g - L^*_{MVC})\} > 0, \tag{21}$$

the corollary follows from Eq. (17). •

It is instructive to examine tha conditions under which the MVCMSE estimator equals the MMSE and MVSE estimators. The MVCMSE estimator collapses to the MVSE estimator when

$$E\{(g - L^*_{MVC})x(\gamma)\} = E\{(g - L^*_{MVC})\}E\{x(\gamma)\}, \tag{22}$$

that is, when the estimation error and the observed data are uncorrelated. The MVCMSE estimator and the MMSE are equivalent under the three conditions that: (i) $E\{(g - L^*_{MVC})^3 x(\gamma)\} = 0$, (ii) either $E\{(g - L^*_{MVC})\} = 0$ or $E\{x(\gamma)\} = 0$, and (iii) $\lambda \neq 2E\{(g - L^*_{MVC})x(\gamma)\}$; or they are equivalent under the two conditions: (i) $\lambda \neq 2E\{(g - L^*_{MVC})x(\gamma)\}$, and (ii) $E\{(g - L^*_{MVC})^3 x(\gamma)\} - \frac{\lambda}{2}E\{(g - L^*_{MVC})\}E\{x(\gamma)\} = 0$.

IV. EXAMPLES

In this section, we consider a series of simple examples to illustrate the theory and to compare MMSE, MVSE, and MVCMSE estimators in several cases of interest. In particular, we examine the estimation of a random variable by a constant, memoryless filtering in additive noise, memoryless filtering in multiplicative noise, and a prediction problem.

The Constant Estimator

We desire to estimate the signal $g(t) = s(t)$ by an estimator of the form

$$\hat{g}(t) = L_g[x(t)] = a, \quad \text{where} \quad I = \phi. \tag{23}$$

Equation(9) yields the expression

$$\begin{aligned} E\{[(s - a^*_{MVC})^2 - E\{(s - a^*_{MVC})^2\}(s - a^*_{MVC})a \\ + \frac{\lambda}{2}[(s - a^*_{MVC}) - E\{(s - a^*_{MVC})\}]a\} = 0. \end{aligned} \tag{24}$$

Since a is a constant, the term involving λ is zero, and the equation to be solved is (for $a \neq 0$)

$$E\{(s - a^*_{MVC})^3 - (s - a^*_{MVC})E\{(s - a^*_{MVC})^2\}\} = 0. \tag{25}$$

The solution is

$$a^*_{MVC} = \frac{E(s^3) - E(s)E(s^2)}{2[E(s^2) - E^2(s)]}, \tag{26}$$

which is the same as the unconstrained MVSE estimator [1]. The MMSE estimator for this problem is from Eq. (6), $a^*_M = E(s)$. If $E(s) = 0$, the variance of the squared error of a^*_M is [1]

$$\begin{aligned} \text{var}\,\{(s - a^*_M)^2\} &= E(s^4) - E^2(s^2) \\ &\geq \text{var}\,\{(s - a^*_{MVC})^2\}, \end{aligned} \tag{27}$$

and the a^*_{MVC} estimator incurs an increase in MSE of $[E\{s^3\}/2E\{s^2\}]^2$. Note that for any zero mean symmetric distribution, $a^*_{MVC} = a^*_{MV} = a^*_M = 0$.

The Memoryless Filter in Additive Noise

Consider the scalar memoryless filtering problem defined by

$$g(t) = T[s(t)] = s(t) \tag{28}$$

$$\hat{g}(t) = L_g[x(\tau)] = a\,x(t) \tag{29}$$

and

$$x(t) = s(t) + n(t), \tag{30}$$

where $s(t)$ and $n(t)$ and zero mean and independent, but not necessarily Gaussian, and

$$E\{s^4(t)\} = (k_s + 1)\sigma_s^4, \quad k_s > 0, \tag{31}$$

$$E\{n^4(t)\} = (k_n + 1)\sigma_n^4, \quad k_n > 0. \tag{32}$$

To begin, we evaluate a^*_M for comparison purposes and find

$$a^*_M = \sigma_s^2/(\sigma_s^2 + \sigma_n^2). \tag{33}$$

The MVSE estimator is the solution to [1]

$$\begin{aligned} k_s\sigma_s^4 + (-3k_s\sigma_s^4 - 2\sigma_s^2\sigma_n^2)a^*_{MV} + (3k_s\sigma_s^4 + 6\sigma_s^2\sigma_n^2)a^{*2}_{MV} \\ + (-k_s\sigma_s^4 - 4\sigma_s^2\sigma_n^2 - k_n\sigma_n^4)a^{*3}_{MV} = 0. \end{aligned} \tag{34}$$

Now, to be specific, we assume that $s(t)$ is Laplacian and $n(t)$ Gaussian, so that $E(s^4) = 6\sigma_s^4 \Rightarrow k_s = 5$, and $E(n^4) = 3\sigma_n^4 \Rightarrow k_n = 2$, and we further assume that $\sigma_s^2 = 1 = \sigma_n^2$. Equation (34) becomes

$$5 - 17a^*_{MV} + 21a^{*2}_{MV} - 11a^{*3}_{MV} = 0 \tag{35}$$

which has two complex conjugate roots and the real root

$$a^*_{MV} \cong 0.5957. \tag{36}$$

Under these same assumptions, the MMSE estimator in Eq. (33) becomes

$$a^*_M = 0.5 \tag{37}$$

with

$$MMSE = MSE(a_M^*) = 0.5 \tag{38}$$

and

$$\text{var}\,\{(s - a_M^* x)^2\} = 0.6875. \tag{39}$$

The values in Eqs. (38) and (39) should be compared to the corresponding results for a_{MV}^*, which are

$$MSE(a_{MV}^*) = 0.5186 \tag{40}$$

and

$$MVSE = \text{var}\,\{(s - a_{MV}^* x)^2\} = 0.6111. \tag{41}$$

Therefore, the unconstrained MVSE estimator achieves an 11.1% reduction in the variance of the squared error with the penalty of a 3.7% increase in MSE.

Using Eq. (9), we find that the MVCMSE estimator under the same assumptions is the solution to

$$-11a_{MVC}^{*3} + 21a_{MVC}^{*2} + (-17 - \lambda)a_{MVC}^* + 5 + \frac{\lambda}{2} = 0 \tag{42}$$

with λ as a parameter, subject to

$$1 - 2a_{MVC}^* + 2a_{MVC}^{*2} = C, \tag{43}$$

with $0.5 < C < 0.5186$. If the desired constraint is a 2% increase in MSE, then $C = 0.51$, and we find from Eq. (43) that $a_{MVC}^* = 0.5707$ or 0.4293. We find by direct substitution that

$$a_{MVC}^* = 0.5707 \tag{44}$$

yields the smaller variance of the squared error, which is

$$\text{var}\,\{(s - a_{MVC}^*)^2\} = 0.6199. \tag{45}$$

The MVCMSE estimator thus provides a 9.8% decrease in the variance of the squared error for a 2% increase in MSE. Note that for this problem, all of the estimators are distinct. However, if $s(t)$ is Gaussian, $a_M^* = a_{MV}^* = a_{MVC}^*$.

The Memoryless Filter in Multiplicative Noise

For this example, $g(t)$ and $\hat{g}(t)$ are as in Eqs (28) and (29), but now the observed data is

$$x(t) = s(t)[1 + n(t)], \tag{46}$$

where we assume that $s(t)$ and $n(t)$ are zero mean and independent. Let us further assume that $\sigma_s^2 = 1$, $\sigma_n^2 = 1$, and that $n(t)$ is Gaussian. The MMSE estimator for this problem is (from Eq. (6)),

$$a_M^* = 0.5 \tag{47}$$

with

$$MMSE = MSE(a_M^*) = 0.5. \tag{48}$$

Now letting $s(t)$ also be Gaussian, we have that

$$\text{var}\,\{(s - a_M^* x)^2\} = 1.515. \tag{49}$$

For these same assumptions, the unconstrained MVSE estimator satisfies

$$13a_{MV}^{*3} - 15a_{MV}^{*2} + 7a_{MV}^{*} - 1 = 0, \tag{50}$$

which has two complex conjugate roots and the root

$$a_{MV}^{*} = 0.2421. \tag{51}$$

The corresponding MSE and MVSE are

$$MSE(a_{MV}^{*}) = 0.633 \tag{52}$$

and

$$MVSE = \text{var}\,\{(s - a_{MV}^{*}x)^2\} = 0.9939. \tag{53}$$

Therefore, the unconstrained MVSE estimator produces a decrease of 34% in the variance of the squared error with a 26.6% increase in MSE.

Under the same assumptions, we find from Eq. (9) that the MVCMSE estimator satisfies

$$13a_{MVC}^{*3} - 15a_{MVC}^{*2} + (7 - \lambda)a_{MVC}^{*} + \frac{\lambda}{2} - 1 = 0 \tag{54}$$

subject to the constraint that

$$1 - 2a_{MVC}^{*} + 2a_{MVC}^{*2} = C. \tag{55}$$

If we wish to restrict the increase in MSE to 20%, we choose $C = 0.6$, which allows us to find two candidate values of a_{MVC}^{*} from Eq. (55) to be 0.7236 or 0.2764. The minimum variance of the squared error is obtained by

$$a_{MVC}^{*} = 0.2764 \tag{56}$$

and has the value

$$\text{var}\,\{(s - a_{MVC}^{*}x)^2\} = 1.2351. \tag{57}$$

Thus, the MVCMSE estimator reduces the variance of the squared error by 18.5% over a_{M}^{*} while allowing an increase in MSE of 20%.

If we assume that $s(t)$ is Laplacian rather than Gaussian, but all of the other assumptions are retained, we find for the MSE, MVSE, and MVCMSE estimators, respectively, that:

$$a_{M}^{*} = 0.5 \tag{58}$$

$$MMSE = MSE(a_{M}^{*}) = 0.5, \tag{59}$$

$$\text{var}\,\{(s - a_{M}^{*}x)^2\} = 2.765; \tag{60}$$

$$a_{MV}^{*} = 0.2723 \tag{61}$$

$$MSE(a_{MV}^{*}) = 0.604 \tag{62}$$

$$\text{var}\,\{(s - a_{MV}^{*}x)^2\} = 2.2983 \tag{63}$$

$$a_{MVC}^{*} = 0.2764, \tag{64}$$

$$MSE(a_{MVC}^{*}) = 0.6 \tag{65}$$

$$\text{var}\,\{(s - a^*_{MVC}x)^2\} = 2.2985. \tag{66}$$

Therefore, in this case the MVSE and MVCMSE estimators are virtually identical.

The Memoryless Predictor

A scalar, memoryless prediction problem is defined by

$$g(t) = T[s(t)] = s(t+\tau),\ \tau > 0, \tag{67}$$

$$\hat{g}(t) = L_g[x(\gamma)] = ax(\gamma),\ I = \{t\}, \tag{68}$$

and

$$x(t) = s(t), \tag{69}$$

where $s(t)$ is zero mean and Gaussian. It is shown in [1] that $a^*_M = a^*_{MV} = R_s(\tau)/R_s(0)$, where $R_s(\tau) = E\{s(t)s(t+\tau)\}$. To find the MVCMSE estimator, we expand and simplify Eq. (9) to produce

$$\begin{aligned} &6R_s(0)R_s(\tau) - 6a^*_{MVC}[2R_s^2(\tau) + R_s^2(0)] + 18a^{*2}_{MVC}R_s(\tau)R_s(0) \\ &\quad -6R_s(0)a^{*3}_{MVC} - 2[R_s(0) - 2a^*_{MVC}R_s(\tau) \\ &+R_s(0)a^{*2}_{MVC}][R_s(\tau) - a^*_{MVC}R_s(0)] + \lambda[R_s(\tau) - a^*_{MVC}R_s(0)] = 0. \end{aligned} \tag{70}$$

Now,

$$a^*_{MVC} = \frac{R_s(\tau)}{R_s(0)} \tag{71}$$

can be seen to solve Eq. (70) by direct substitution. To see that a^*_{MVC} must be the absolutely minimizing solution let

$$L_1[x(\gamma)] = a_1 x(t) = a_1 s(t) \tag{72}$$

be the absolutely minimizing functional and write

$$a_1 s(t) = a^*_{MVC}s(t) - a_2 s(t). \tag{73}$$

Then

$$\begin{aligned} \text{var}\,\{(s(t+\tau) - a_1 s(t))^2\} = {} & \text{var}\,\{(s(t+\tau) - a^*_{MVC}s(t))^2\} \\ & - 2\lambda[R_s(\tau) - a^*_{MVC}R_s(0)]a_2 + a_2^2 E\{(s(t+\tau) \\ & - a^*_{MVC}s(t))^2 s^2(t)\} - a_2^2 E\{(s(t+\tau) - a^*_{MVC}s(t))^2\}\cdot \\ & E\{s^2(t)\} + A, \end{aligned} \tag{74}$$

where A is a term greater than or equal to zero. For a^*_{MVC} in Eq. (71), the term involving λ drops out, and the next to last term is greater or equal to zero, since for Gaussian random variables, $E(X^2Y^2) \geq E(X^2)E(Y^2)$. If either of the last two terms on the right of Eq. (74) is strictly positive, then we have a contradiction, which implies $a_2 = 0$. Therefore, $a_1 = a^*_{MVC}$ is the absolutely minimizing parameter, and hence, for the scalar, Gaussian prediction problem, $a^*_{MVC} = a^*_{MV} = a^*_M$.

V. CONCLUSIONS

The minimum variance of the squared error estimation problem subject to a constraint on the error variance has been posed and conditions which the optimal (MVCMSE) estimator must satisfy have been derived. Several examples have illustrated that the MMSE, MVSE, and MVCMSE estimators are often distinct and that clear performance tradeoffs are involved. Interestingly, in some simple cases, the constraint greatly simplifies the solution of the problem.

ACKNOWLEDGEMENTS

This research was supported, in part, by the Statistics and Probability program, Office of Naval Research under Contract No. N00014-81-K-0299. The author wishes to thank Professor Michael K. Sain of the University of Notre Dame for discussions concerning this work and for providing copies of References [2] and [3].

REFERENCES

[1] M. K. Sain and C. R. Souza,, "A Projection Principle for Minimum Variance Estimation," Proceedings of the Nat'l. Electronics Conf., Vol. XXII, pp. 683-686, 1966.

[2] M. K. Sain and C. R. Souza, "Linear, Minimum Variance, Squared Error Estimation: Motivation and Applications," Dept. of Elec. Eng., Univ. of Notre Dame, Notre Dame, IN, Tech, Rpt. EE-6710, April 3, 1967.

[3] M. K. Sain and C. R. Souza, "A theory for linear estimators minimizing the variance of the error squared," IEEE Trans. Inform. Theory, Vol. IT-14, pp. 768-770, Sept. 1968.

[4] M. K. Sain, "Control of Linear Systems According to the Minimal Variance Criterion - A New Approach to the Disturbance Problem," IEEE Trans. Automat. Control, Vol. AC-11, pp. 118-122, Jan. 1966.

ITERATIVE MAXIMUM-LIKELIHOOD ESTIMATION OF TOEPLITZ CONSTRAINED COVARIANCES

MICHAEL I. MILLER, DONALD L. SNYDER, MICHAEL TURMON

Department of Electrical Engineering and
The Institute for Biomedical Computing
Washington University
St. Louis, Missouri 63130

As first shown by Burg, when given exact values of some finite set of the autocovariances of a stationary time-series, but no values of the actual time series, the maximum entropy density becomes a multivariate Gaussian constructed by maximizing the entropy function subject to covariance constraints.[1,2] For the spectrum estimation problem we now address, samples from the actual time series do exist, whereas the autocovariance values themselves are unknown. In order to apply the entropy theory, many investigators have generated second order statistics from the time series, from which moment constraints are assumed.[3] We instead derive maximum-likelihood estimates of the autocovariance parameters under the Gaussian model. This involves the use of the expectation maximization algorithm[4] which depends on the construction of a complete-incomplete data set-up in which the incomplete data correspond to the finite length measured sequence and the complete data correspond to the infinite duration sequence.[5] We proceed as follows. We assume that we are given an observed series $\{y_0,\cdots,y_{G-1}\}$ of length G, which is part of a larger stationary, zero-mean Gaussian periodic time-series of period N>G where $y_0=y_N$, $y_1=y_{N+1},\cdots$, and we wish to find the maximum-likelihood estimates of the Toeplitz covariance matrix K of the N-periodic sequence. The choice of the finite period of length N is arbitrary, and we can allow it to grow when modeling stationary non-periodic processes. We solve for the maximum-likelihood estimates $\hat{K}$ via an iterative maximization. The proper choice for the *complete-data* becomes the entire set of samples $\{y_0,\cdots,y_{G-1},$ $y_G,\cdots,y_{N-1}\}$ from the N-periodic time-series. We denote this complete-data by the N-dimensional vector Y_N, consisting of the given G-dimensional vector Y_G corresponding to the set $\{y_0,\cdots,y_{G-1}\}$, augmented by the N-G dimensional vector Y_A corresponding to the set $\{y_G,y_{G+1},\cdots,y_{N-1}\}$. The complete-data density is

$$f(Y_N;K) = (2\pi)^{-\frac{N}{2}} \det^{-\frac{1}{2}}|K| \exp(-\frac{1}{2} Y_N^{\dagger} K^{-1} Y_N) , \qquad (1)$$

with † denoting hermitian transpose, det | | denoting matrix determinant, and K^{-1} matrix inverse. We rewrite (10) as

$$f(Y_N;v) = = \frac{\exp(-\sum_{k=0}^{N-1} v_k[\sum_{m=0}^{N-1} y(m)y^*(<m+k>_N)])}{\exp(-\sum_{k=0}^{N-1} v_k[\sum_{m=0}^{N-1} y(m)y^*(<m+k>_N)])dY_N} , \qquad (2)$$

where $< >_N$ denotes *modulo* N, and * denotes complex conjugate. Equation 2 is derived by defining the N-squared circulant matrix $V=\frac{1}{2}K^{-1}$ whose entries are $V_{ij}=v_{j-i}$ with $v_{j-i}=v^*_{i-j}, v_{j-i}=v^*_{N-(j-i)}$. Since V is related to the inverse of K, finding the maximum-likelihood estimate of V yields the maximum-likelihood estimates of K. As we have shown in a previous paper,[6] the maximum-likelihood estimates $\hat{K}$ maximize the conditional expectation of the complete-data log-likelihood $E\{\log f(Y_N;v)|Y_G,\hat{K}\}$. Maximizing the conditional-mean of the log-likelihood yields the following equation which $\hat{K}$ must satisfy:

$$\hat{K}(\tau) = \sum_{m=0}^{N-1} E\{y(m)y^*(\langle m+\tau\rangle_N)\,|Y_G,\hat{K}\} \text{ .} \tag{3}$$

Iteratively maximizing (3) yields the sequence of iterates $\{K^{(p)}(\tau);$ $\tau=1,2,\cdots\}$ given by

$$K^{(p+1)}(\tau) = \sum_{m=0}^{N-1} E\{y(m)y^*(\langle m+\tau\rangle_N)\,|Y_G,K^{(p)}\} \text{ .} \tag{4}$$

This iteration is an instance of an E-M algorithm. We have shown that if the iteration of (4) converges, the convergence point satisfies the general necessary conditions of Burge et al.[7] for a Toeplitz constrained maximum-likelihood estimator.

References

1. J. P. Burg, *Maximum Entropy Spectral Analysis*, Univ. Microfilms No. 75-25, Stanford University, Stanford, California, 1975.

2. E. T. Jaynes, "On the Rationale of Maximum-Entropy Methods," Proc. of the IEEE, vol. 70, pp. 939-952, September 1982.

3. J. P. Burg, *Maximum Entropy Spectral Analysis*, Univ. Microfilms No. 75-25, Stanford University, Stanford, California, 1975.

4. A. D. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood From Incomplete Data Via the EM Algorithm," J. of the Royal Statistical Society, vol. B.39, pp. 1-37, 1977.

5. M. I. Miller and D. L. Snyder, "The Application of Maximum-Entropy and Maximum-Likelihood for the Solution of Incomplete and Noisy Data Problems in Estimating Point-Process Intensities, Probability Densities, and Spectral Densities," Proceedings of the IEEE (in review).

6. M. I. Miller and D. L. Snyder, "Iterative Algorithms and Architectures for Maximum-Entropy and Its Relation to Maximum-Likelihood With Application to Imaging," SPIE 30th Annual International Technical Symposium on Optical, San Diego, CA, 1986.

7. J. P. Burg, D. G. Luenberger, and D. L. Wenger, "Estimation of Structured Covariance Matrices," Proc. of the IEEE, vol. 70,9, pp. 963-974, September 1982.

ADAPTIVE FILTERING FOR A LOCALIZED DISTORTION MEASURE

JERRY D. GIBSON AND STEVEN D. GRAY

Department of Electrical Engineering
Texas A&M University
College Station, TX 77843

Several adaptive signal processing problems can be represented by the structure shown in Fig. 1, where $P(z)$ is a polynomial transfer function and $\{n_j\}$ is an additive, zero mean, independent noise sequence. The input sequence to $P(z)$ and $\hat{P}(z)$, $\{x_j\}$, is assumed to be white and independent of n_j. The coefficients of the polynomial $\hat{P}(z)$ are to be adjusted to minimize some statistical measure of the error ϵ_j. The most common distortion criterion is the mean squared value of the error (MSE), denoted $E[\epsilon_j^2]$, and the familiar Widrow–Hoff LMS algorithm [1] uses an instantaneous estimate of the gradient of $E[\epsilon_j^2]$ to produce the parameter adaptation rule

$$\mathbf{W}_{j+1} = \mathbf{W}_j + 2\mu\epsilon_j\mathbf{X}_j, \tag{1}$$

where $\mathbf{W}_j$ is the vector of coefficients of $\hat{P}(z)$, $\mathbf{X}_j$ is a vector of past inputs (x_j), and μ is a convergence factor to be chosen. Recently, a related class of algorithms was proposed which approximately minimizes $E[\epsilon_j^{2K}]$ by using an instantaneous estimate of the gradient of this quantity to obtain the algorithm [2,3]

$$\mathbf{W}_{j+1} = \mathbf{W}_j + 2\mu K\epsilon_j^{2K-1}\mathbf{X}_j. \tag{2}$$

For $\hat{P}(z)$ a transversal filter form of the adaptive linear combiner, it is shown in [2] that depending upon the noise (n_j) distribution, the LMF$(K = 2)$ algorithm may yield a smaller mean squared coefficient error than the LMS algorithm.

Motivated by the work of Sain in stochastic control [4], we wish to design an adaptive gradient algorithm which minimizes the variance of ϵ_j^2 subject to a constraint on the MSE. Thus, we are attempting to minimize the variation of the squared error about some acceptable value, say σ_o^2, which is, of course, greater than the minimum MSE possible. Therefore, we can guarantee that the performance does not deviate too much from our acceptable distortion value. Posing and solving the variational problem and approximating the expectations involved by instantaneous estimates produces the algorithm [5]

$$\mathbf{W}_{j+1} = \mathbf{W}_j + 4\mu\epsilon_j^3\mathbf{X}_j - 2\mu(2\sigma_o^2 + \lambda)\epsilon_j\mathbf{X}_j \tag{3}$$

where λ is an undetermined multiplier and σ_o^2 is the acceptable value of $E[\epsilon_j^2]$. We denote the adaptation rule in Eq. (3) as the least variance subject to a constraint on mean squared error (LVCMS) algorithm. Comparing Eqs. (1)–(3), it is interesting to note that the correction term for the new LVCMS algorithm is a combination of the correction terms for the LMS and LMF algorithms, with a different weight on the LMS term. A bound on the convergence parameter μ in Eq. (3) can be derived and is given by [5]

$$0 < \mu < \frac{1}{[6E(n_j^2) - 2\sigma_o^2 - \lambda]\, tr\ \mathbf{R}} \tag{4}$$

where $\mathbf{R} = E\{\mathbf{X}_j\mathbf{X}_j^T\}$ and "tr" denotes the trace of a matrix. An often-used indicator of performance for adaptive filtering algorithms is the misadjustment (M) which for the LVCMS algorithm can be shown to be [5]

$$M = \frac{[4E(n_j) - 4(2\sigma_o^2 + \lambda)E(n_j^4) + (2\sigma_o^2 + \lambda)^2E(n_j^2)]\mu N E(x_j^2)}{[6E(n_j^2) - (2\sigma_o^2 + \lambda)]E(n_j^2)}. \tag{5}$$

Letting

$$P(z) = 0.1+0.2z^{-1}+0.3z^{-2}+0.4z^{-3}+0.5z^{-4}+0.4z^{-5}+0.3z^{-6}+0.2z^{-7}+0.1z^{-8}, \quad (6)$$

and setting the time constants of the LMS, LMF, and LVCMS algorithms to 555 samples, the theoretical and experimental misadjustment values can be contrasted for each of several noise densities. For the simulations, each coefficient of $\hat{P}(z)$ is initialized by adding a zero mean, Gaussian random variate with standard deviation 0.75 to the true values, and 20 sequences of length 2500 samples were averaged. In the following, the theoretical misadjustment value is listed first, followed by the experimental result in parentheses. For a uniform density on $\{n_j\}$ and $\{x_j\}$, the LMS, LMF, and LVCMS algorithms have the misadjustment values .0081(.0089), .0034(.00415), and .00252(.00303), respectively. The corresponding values for Gaussian densities are .0081(.00905), .0135(.0159), and .00925(.0102), and for Laplacian densities, .0081(.00912), .081(.101), and .02548(.0308). There is good agreement between theoretical and experimental values, with any discrepancies due to the assumptions used to derive the theoretical misadjustments [2,5]. Note that for the uniform density, the LVCMS algorithm has the smallest misadjustment followed in order by the LMF and LMS algorithms. With the heavier-tailed Gaussian and Laplacian densities, the LMS algorithm has the smallest misadjustment followed by the LVCMS and LMF algorithms.

Further simulation results indicate that the LVCMS algorithm yields the smallest variance of the squared error followed in order by the LMS and LMF adaptation rules. The LMS algorithm always has the smallest mean squared estimation error. The LMF algorithm has difficulties converging with the Gaussian and Laplacian densities.

ACKNOWLEDGEMENT

This work was supported, in part, by the Statistics and Probability Program, Office of Naval Research under Contract No. N00014-81-K-0299.

REFERENCES

[1] B. Widrow and S. D. Stearns, Adaptive Signal Processing, Prentice-Hall, Inc. Englewood Cliffs, NJ, 1985.

[2] E. Walach and B. Widrow, "The Least Mean Fourth (LMF) Adaptive Algorithm and Its Family," IEEE Trans. Inform. Theory, Vol. IT-30, pp. 275-283, March 1984.

[3] E. Walach, "On High-Order Error Criteria for System Identification," IEEE Trans. Acoust., Speech, and Signal Processing, Vol. ASSP-33, pp. 1634-1635, Dec. 1985.

[4] M. K. Sain, "Control of Linear Systems According to the Minimal Variance Criterion - A New Approach to the Disturbance Problem," IEEE Trans. Automatic Control, Vol. AC-11, pp. 118-121, Jan. 1966.

[5] J. D. Gibson and S. D. Gray, "MVSE Adaptive Filtering Subject to a Constraint on MSE," submitted to IEEE Trans. Circuits and Systems, August, 1986.

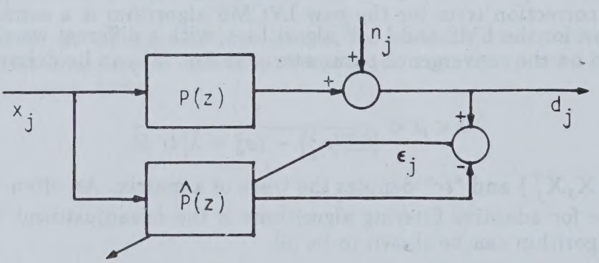


Fig. 1. Adaptive Filtering Configuration

LATTICE IMPLEMENTATION of a NOVEL RECURSIVE ESTIMATION ALGORITHM

Y.F. HUANG and ASHOK K. RAO
Department of Electrical & Computer Engineering
University of Notre Dame
Notre Dame, IN 46556

ABSTRACT

In this paper, the lattice structure implementation of a new recursive estimation algorithm for AR and ARX models is presented. The lattice algorithms incorporate information-dependent updating and forgetting factor. The order of computation of the lattice algorithms is the same as that of the direct implementation. Simulation results show that the algorithm has a fast rate of convergence and low mean squared output error.

1. INTRODUCTION

In recent years, lattice-filter implementations of adaptive algorithms have become popular for a number of reasons. Among others, the most prominent ones are: (1) The modular structure of lattice filters which renders them particularly suitable for VLSI implementations; (2) The low sensitivity of the lattice coefficients to numerical perturbation and; (3) The fact that the lattice coefficients are independent of the filter order thus making it possible to add successive lattice stages or subtract existing ones without recalculating the already existing coefficients.

Most of the work in the area of adaptive lattice algorithms has been centered on the development, refinement and application of the recursive least-squares lattice (LSL) algorithms . The LSL algorithms which were first presented by Morf, Viera and Lee in 1977 [1], provide an exact solution to the least-squares problem at each time step. An alternate class of lattice algorithms, the gradient adaptive lattice (GAL) algorithms, which are based on gradient search techniques, was introduced by Griffiths [2]. Though the computational complexity of the GAL algorithms is less than that of the LSL algorithms, their performance is generally inferior to the LSL in some applications. However, in some applications like ADPCM,the GAL algorithms perform as well as the LSL [3].

Lattice implementations of estimation algorithms which attempt to minimize criteria other than the norm of the prediction error are scarce. In this paper we present a lattice implementation of the optimal bounding- ellipsoid (OBE) algorithm [4]. The OBE algorithm is a recursive parameter estimation algorithm which incorporates data-dependent updating and is applicable to autoregressive processes with auxiliary inputs and bounded noise (ARX processes). The key feature of the OBE is that at every time step a decision is made whether or not to update the parameter estimates. This decision is based on the information content of the incoming data sample. The OBE , on the average, updates only 20 % of the time . This fact makes the OBE eminently suitable for processing multiple channels simultaneously, as it makes a time-sharing type signal processor feasible. Another advantageous feature of the OBE algorithm is that after the algorithm converges the parameter updating stops. This is particularly useful in control applications where non-cessation of parameter updating may excite the high frequency modes of the system thus causing instability [5]

The lattice implementation of the OBE algorithm involves the same order of complexity , O(N^2), per time step, as its direct implementation. In addition, it has all the advantages of the lattice structure discussed previously. We shall first present an outline of the OBE algorithm and then the lattice implementation of the algorithm

2. OBE ALGORITHM

Consider the ARX model

$$y(k) = a_1 y(k-1) + a_2 y(k-2) + \ldots a_L y(k-L) + b_1 u(k-1) + b_2 u(k-2) + \ldots$$
$$\ldots + b_M u(k-M) + v(k) \qquad (1)$$

where y(k) is the measured output of the process, u(k) is the measurable input and v(k) is the noise at time k,which is assumed to be zero mean white noise. The above equation can be recast as :

$$y(k) = \theta^{*T} \Phi(k)$$

where

$$\theta^* = [a_1 , a_2 \ldots a_L , b_1 , b_2 , \ldots. b_M]^T$$

is the vector of true parameters and

$$\Phi(k) = [y(k-1), y(k-2), .. y(k-L), \quad u(k-1), u(k-2), .. u(k-M)]^T$$

is the regressor vector.

A key assumption made now is that the noise is bounded in magnitude, i.e.

$$v(k)^2 < \gamma^2 \text{ for all k, thus}$$
$$(y(k) - \theta^{*T} \Phi(k))^2 < \gamma^2$$

Let S_k be a subset of $\Re^{L+M}$ defined by

$$S_k = \{ \theta : (y(k) - \theta^T \Phi(k))^2 < \gamma^2 , \theta \varepsilon \Re^{L+M}$$

It is seen that, from a geometric point of view, S_k is a convex polytope in the parameter space and contains the vector of true parameters.

The OBE algorithm starts off with a large ellipsoid, E_0, in $\Re^{2N}$ which contains all possible values of the modelled parameter θ^*. After the first observation y_1 is acquired, an ellipsoid is found which bounds the intersection of E_o and the convex polytope S_1.This ellipsoid must be optimal in some sense, say minimum volume or by any other criterion, to assure convergence. Denoting the optimal ellipsoid by E_1, one can proceed exactly as before with the future observations and obtain a sequence of optimal bounding ellipsoids $\{ E_k \}$. The center of the ellipsoid E_k is taken as the parameter estimate at the k-th instant and is denoted by θ_k. It can be shown that all the ellipsoids $\{ E_k \}$ contain the true parameter θ^*, provided that E_0 does The time update equations of the OBE algorithm are very similar to those of the weighted recursive least squares algorithm. Details of the algorithm can be found in [4] and [9].

3. AR LATTICE

Consider the following well-known RLS lattice recursions for an AR model [6]

$$e_m(t) = e_{m-1}(t) - k^b_{m-1}(t-1) r_{m-1}(t-1) \qquad (2)$$

$$r_m(t) = r_{m-1}(t-1) - k^f_{m-1}(t-1)\, e_{m-1}(t) \qquad (3)$$

where, $e_0(t) = r_0(t) = y(t)$, $e_m(t)$ is the forward prediction error of order m, $r_m(t)$ is the backward prediction error of order m and k^b_m and k^f_m are, respectively, the m-th backward and forward partial correlation (PARCOR) coefficients.Iterating Eq.(2) up to order m = N yields

$$e_N(t) = y(t) - k^b_0(t-1)\, r_0(t-1) - k^b_1(t-1)\, r_1(t-1) - \ldots - k^b_{N-1}(t-1)\, r_{N-1}(t-1)$$

Thus the N-th order predictor of y(t) is

$$\hat{y}(t/N) = k^b_0(t-1)\, r_0(t-1) + k^b_1(t-1)\, r_1(t-1) + \ldots + k^b_{N-1}(t-1)\, r_{N-1}(t-1) \qquad (4)$$

Compare Eq.(4) to the equation for a transversal predictor given by

$$\hat{y}(t/N) = \hat{a}_1(t-1)\, y(t-1) + \hat{a}_2(t-1)\, y(t-1) + \ldots\ \hat{a}_N(t-1)\, y(t-N)$$

Now, defining $y_t = (0,\ldots y(0), y(1),\ldots y(t))$, $r_{m,t} = (0, \ldots r_m(0), r_m(1),\ldots r_m(t))$, the optimal predictor for the RLS transversal case can be thought of in geometrical terms as the projection of y_t onto the regressor space spanned by $y_{t-1}, y_{t-2}, \ldots\ y_{t-N}$ .The backward errors $r_{0,t-1}, r_{1,t-1}, \ldots r_{N-1,t-1}$ span exactly the same space with the difference that they are mutually orthogonal [6], and the optimal predictor for the RLS lattice case is again the projection of y_t onto the above regressor space. Thus the predictors for the RLS lattice and transversal case are identical.

In general, the OBE estimates at every time step are not identical to the RLS estimates. However after the transient stage they will be close enough. This seems to justify the imposition of the lattice structure (Eq.(2) and Eq.(3)) on the forward and backward errors of different orders for the OBE algorithm. In principle therefore, the OBE algorithm can be used to calculate the transversal filter coefficients and a step down procedure [6] can be then used to calculate the optimal PARCOR coefficients. However this method will involve an excessive amount of computation. Furthermore, many of the advantageous features of the lattice structure mentioned earlier will be lost since the transversal filter coefficients are being used to obtain the lattice coefficients. For example additional lattice stages cannot be added or existing stages subtracted with the same ease since all the transversal filter coefficients change when the order is changed. The lattice coefficients will be more sensitive to numerical perturbations. We therefore prefer applying the OBE in such a way that at every time step, the estimate of the PARCOR coefficients is obtained directly. This problem can be tackled by working in the space of PARCOR coefficients instead of the space of the transversal filter coefficients. More specifically we define the ellipsoid

$$E'_0 = \{ \theta' : \theta'^T P_0^{-1} \theta < \frac{1}{\varepsilon^2} \}$$

and the convex polytope

$$S'_t = \{\theta' : (y(t) - \theta'^T \Phi'(t))^2 \le \gamma^2\}$$

where

$\theta' = (k^b_0, k^b_1, \ldots k^b_{N-1})^T$ and $\Phi'(t) = (r_0(t-1), r_1(t-1), \ldots r_{N-1}(t-1))^T$. Further P_0

= I and ε is a small number. If the true PARCOR coefficients are defined to be the coefficients obtained by applying the step down procedure to the true AR parameters of the system, then it is clear that each one of the convex polytopes S'_t, t = 1 ..T , will contain the true backward PARCOR coefficient vector. The OBE algorithm can now be applied with the PARCOR coefficient vector θ' set equal to the parameter vector θ and the vector of backward errors $\Phi'(t)$ set equal to the regressor vector $\Phi(t)$ thus yielding the the time update for the backward PARCOR coefficients. However, it is clear from Eq.3 that in order to obtain the backward errors of different orders at time t, the forward PARCOR coefficients at time t-1 are required. The time update equations for the forward coefficients can be obtained as follows.

Iterating Eq.(3) yields

$$r_N(t) = y(t-N) - k^f_0(t-N)\, e_0(t-N+1) - k^f_1(t-1)\, e_1(t-N+2) \;\ldots - k^f_{N-1}(t-1)\, e_{N-1}(t)$$

Since the backward errors are expected to be bounded, one can therefore define a convex polytope in the space of the forward PARCOR coefficients as

$S''(t) = (\, \theta'' : (\, y(t-N) - \theta''^T\Phi''(t)\,)^2 \leq \gamma'^2\,)$

where

$$\theta'' = (\, k^f_0 , k^f_1, \ldots k^f_{N-1}\,)^T \text{ and } \Phi''(t) = (\, e_0(t-N+1), e_1(t-N+2), \ldots e_{N-1}(t)\,)^T$$

γ'^2 is set higher than γ^2 though its exact value is not critical since the algorithm is relatively insensitive to the value of the noise bound. We can now apply the OBE algorithm with $\theta = \theta''$ and $\Phi = \Phi''$ and obtain the time updates of the forward PARCOR coefficients.

To summarize, the algorithm for the AR lattice starts off by initializing the forward and backward PARCOR coefficients to zero. The backward error r_m , $0 \leq m \leq N-1$ is initialized to a small constant. At time t, the lattice recursions Eq.(2) and Eq.(3) are used to generate the forward and backward errors of all orders upto N. The OBE algorithm is applied to estimate the backward and forward coefficients separately, thus updating the coefficients in time. An important point to be noted is that for the backward error recursions at time t, we need $k^f_0(t-1), k^f_1(t-1) \ldots k^f_{N-1}(t-1)$. However, the OBE time update at time t-1 has made available the coefficients $k^f_0(t-N), k^f_1(t-N+1) \ldots k^f_{N-1}(t-1)$. We therefore have to approximate the former set of coefficients by the latter set. This is a valid approximation because if N is small , then $k_m(t-1)$ is approximately equal to $k_m(t-N+m)$, as verified by simulation.

The algorithmic complexity of the lattice implementation is twice that of the direct form because the OBE algorithm is applied two times at every iteration. The order of computation is still $O(N^2)$. For the stationary case, since the backward and forward coefficients are expected to be equal, the OBE needs to be applied only once to the forward error (i.e. to the backward coefficients), and thus the algorithmic complexity is the same as that of the direct implementation.

4. ARX LATTICE

Consider the ARX model

$$y(t) = a^*_1\, y(t-1) + \ldots a^*_n\, y(t-n) + b^*_1 u(t-1) + \ldots b^*_n u(t-n) + v(t) \tag{5}$$

where a^* and b^* denote the true coefficients of the model.

For this ARX model, the lattice structure can be developed by either embedding the ARX model into a two channel AR model of the joint input / output process [7], or by deriving the lattice structure for the multi-dimensional case from scratch [8]. The RLS lattice recursions for ARX models are -

$$\begin{bmatrix} e^y{}_m(t) \\ e^u{}_m(t) \end{bmatrix} = \begin{bmatrix} e^y{}_{m-1}(t) \\ e^u{}_{m-1}(t) \end{bmatrix} - \begin{bmatrix} k^b{}_{11,m-1}(t-1) & k^b{}_{12,m-1}(t-1) \\ k^b{}_{21,m-1}(t-1) & k^b{}_{22,m-1}(t-1) \end{bmatrix} \begin{bmatrix} r^y{}_{m-1}(t-1) \\ r^u{}_{m-1}(t-1) \end{bmatrix}$$

$$\begin{bmatrix} r^y{}_m(t) \\ r^u{}_m(t) \end{bmatrix} = \begin{bmatrix} r^y{}_{m-1}(t-1) \\ r^u{}_{m-1}(t-1) \end{bmatrix} - \begin{bmatrix} k^f{}_{11,m-1}(t-1) & k^f{}_{12,m-1}(t-1) \\ k^f{}_{21,m-1}(t-1) & k^f{}_{22,m-1}(t-1) \end{bmatrix} \begin{bmatrix} e^y{}_{m-1}(t) \\ e^u{}_{m-1}(t) \end{bmatrix} \tag{6}$$

and $e^y{}_0(t) = r^y{}_0(t) = y(t)$; $e^u{}_0(t) = r^u{}_0(t) = u(t)$.

$e^y{}_m(t)$ and $e^u{}_m(t)$ are the forward errors; $r^y{}_m(t)$ and $r^u{}_m(t)$ are the backward errors, of order m, in y and u respectively. The PARCOR coefficients now are 2 X 2 matrices.

If the input is white, then recursion for $e^u{}_m(t)$ is superfluous and, in addition, it can be shown that the coefficients $k^b{}_{21,m-1}(t-1)$, $k^b{}_{22,m-1}(t-1)$, $k^f{}_{12,m-1}(t-1)$, $k^f{}_{22,m-1}(t-1)$ are zero. Let us denote the backward coefficient matrix of order m by K_m (a 2 x 2 matrix), and the forward coefficient matrix of order m by $K^*{}_m$. Then from [8] , we have

$$K_m(t) = F^T{}_m(t)\, [\, R^r{}_m(t-1)\,]^{-1} \tag{7}$$

where

$$F_m(t) = \sum_{k=1}^{t} \lambda^{t-k} \begin{bmatrix} y(k-m-1) \\ u(k-m-1) \end{bmatrix} \begin{bmatrix} e^y_m(k) & e^u_m(k) \end{bmatrix} \tag{8}$$

and

$$R^r_m(t) = \sum_{k=1}^{t} \lambda^{t-k} \begin{bmatrix} r^y_m(k) \\ r^u_m(k) \end{bmatrix} \begin{bmatrix} r^y_m(k) & r^u_m(k) \end{bmatrix} \tag{9}$$

and $\lambda \leq 1$ is the forgetting factor or adaptation constant.

Now $e^u{}_m(k) = u(k) - u(k / m)$, where $u(k / m)$ is the predicted value of u(k) based on m previous samples u(k-1).. u(k-m) . However if u(k) is white, there is no way that u(k) can be predicted on the basis of previous measurements of u. Hence $u(k/m) = 0$, and so $e^u_m(k) = u(k)$, for all m and k. Now since u(k) is uncorrelated with past values of the input u, and the output y, the time averaged correlation between y(k-m-1) and u(k) will be zero and the same is true for the time averaged correlation between u(k-m-1) and $e^u_m(k) = u(k)$. Therefore the second column of the matrix $F_m(t)$ is zero and so $F_m(t)$ has the form

$$F_m(t) = \begin{bmatrix} * & 0 \\ * & 0 \end{bmatrix} \tag{10}$$

where * denotes the non-zero elements.

Unlike the forward predictor, the backward predictor of u(k-m), i.e. $u_b(k/m)$ is not equal to zero, because the value of u(k-m) can be predicted by examining future values of the output - y(t-m+1), y(t-m+2) ... y(t). Therefore $r^u_m(k) = u(k)$ for m = 0 only. From Eq.(9), one can see that in general all elements of the matrix $R^r_m(t-1)$, will be non-zero. Hence from Eq.(7) and Eq.(10), $K_m(t)$ has the form

$$\begin{aligned} K_m(t) &= \begin{bmatrix} * & * \\ 0 & 0 \end{bmatrix} \begin{bmatrix} * & * \\ * & * \end{bmatrix} \\ &= \begin{bmatrix} * & * \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{11}$$

The forward PARCOR coefficient of order m, denoted by $K^*_m(t)$ (a 2 X 2 matrix), is given by

$$K^*_m(t) = F_m(t)\,[R^f_m(t)]^{-1} \tag{12}$$

where

$$R^f_m(t) = \sum_{k=1}^{t} \lambda^{t-k} \begin{bmatrix} e^y_m(k) \\ e^u_m(k) \end{bmatrix} \begin{bmatrix} e^y_m(k) & e^u_m(k) \end{bmatrix} \tag{13}$$

Since $e^u_m(k) = u(k)$, and since the input u(k) is uncorrelated with y(k) and with previous values of y and u, this would mean that $R^f_m(t)$ has the form

$$R^f_m(t) = \begin{bmatrix} * & 0 \\ 0 & * \end{bmatrix} \tag{14}$$

Hence from Equs. (10), (12), and (14), $K^*_m(t)$ has the form

$$K^{*}_{m}(t) = \begin{bmatrix} * & 0 \\ * & 0 \end{bmatrix} \qquad ...(15)$$

Thus the second column of $K^{*}_{m}(t)$ is zero for all m and t . In the ensuing discussion, we assume that the input u is white so as to simplify the problem. However the more general correlated input case can be handled with only slight modifications of the algorithm.

The above structure of the PARCOR coefficient matrices holds, strictly speaking, only when the RLS estimation algorithm is used. But the OBE algorithm can be expected to yield similar estimates, and so we will impose this structure on the PARCOR coefficient estimates obtained by applying the OBE algorithm.

From Eq.(6), we have

$$e^{y}_{m}(t) = e^{y}_{m-1}(t) - k^{b}_{11,m-1}(t-1)\, r^{y}_{m-1}(t-1) - k^{b}_{12,m-1}(t-1)\, r^{u}_{m-1}(t-1)$$

Iterating this equation up to order n, gives

$$e_{n}(t) = y(t) - k^{b}_{11,0}(t-1)\, r^{y}_{0}(t-1) \quad ...- k^{b}_{11,n-1}(t-1)\, r^{y}_{n-1}(t-1) - k^{b}_{12,0}(t-1)\, r^{u}_{0}(t-1)$$
$$... \qquad ... - k^{b}_{12,n-1}(t-1)\, r^{u}_{n-1}(t-1) \qquad (16)$$

Just as in the AR case we can formulate the convex polytope

$$S_{1}(t) = \{\theta_{1} : (y(t) - \theta_{1}^{T}\Phi_{1}(t))^{2} \leq \gamma_{1}^{2}\}$$

where

$$\theta_{1} = (k^{b}_{11,0}, ...\, k^{b}_{11,n-1},\, k^{b}_{12,0}, ...\, k^{b}_{12,n-1})^{T} \text{ and}$$
$$F_{1}(t) = (r^{y}_{0}(t-1), r^{y}_{1}(t-1), ...\, r^{y}_{n-1}(t-1),\, r^{y}_{0}(t-1), r^{u}_{1}(t-1), ...\, r^{u}_{n-1}(t-1))^{T}$$

The OBE algorithm is now used to update the coefficients. The estimate at the t-th iteration - $\theta(t)$, in the OBE equations is set equal to the estimate of the backward PARCOR coefficients at the t-th iteration - $\theta_{1}(t)$ and the regressor vector $\Phi(t)$ in the OBE algorithm is set equal to $\Phi_{1}(t)$.

For the backward errors since
$k^{f}_{12,m}(t-1) = k^{f}_{22,m}(t-1) = 0$,

we have the following equations

$$r^{y}_{m}(t) = r^{y}_{m-1}(t-1) - k^{f}_{11,m-1}(t-1)\, e^{y}_{m-1}(t) \text{ , and}$$
$$r^{u}_{m}(t) = r^{u}_{m-1}(t-1) - k^{f}_{21,m-1}(t-1)\, e^{y}_{m-1}(t) \qquad (17)$$

Iterating Eq.(17) up to order n yields,

$$e_{n}(t) = r^{y}_{0}(t-n) - k^{f}_{11,0}(t-n)\, e^{y}_{0}(t-n+1) \quad ...- k^{f}_{11,n-1}(t-1)\, e^{y}_{n-1}(t)$$

Since $r^y_0(t-n) = y(t-n)$, we can formulate the convex polytope

$$S_2(t) = \{\theta_2 : (y(t-n) - \theta_2^T \Phi_2(t))^2 \leq \gamma_2^{\,2}\}$$

where

$$\theta_2 = (k^f_{11,0}, k^f_{11,1}, \ldots k^f_{11,n-1})^T \text{ and}$$

$$\Phi_2(t) = (e^y_0(t-n+2), e^y_1(t-n+2), \ldots e^y_{n-1}(t))^T$$

(18)

Using Eq.(18) to define the estimate vector and the regressor vector, the OBE algorithm can now be applied to update the coefficients.

For the remaining coefficients $k^f_{21,m}$, $m = 0 \ldots n-1$, we proceed exactly as above by first iterating the appropriate order recursive equation up to order n, and then defining a convex polytope in the appropriate space. Finally the OBE equations are applied with $\theta\hat{}(t) = (k^f_{11,0}(t-n), k^f_{11,1}(t-n+1), \ldots k^f_{11,n-1}(t-1))^T$, $\Phi(t) = \Phi_2(t)$, and $\gamma^2 = \gamma_2^{\,2}$

The values of $\gamma_1^{\,2}$ $\gamma_2^{\,2}$ are set higher than the bound γ^2, as the exact value of the bound on the backward errorsis usually unknown. Furthermore, as before we face the problem that the OBE at time t-1, does not yield the right set of coefficients for the lattice recursions for the backward errors at time t. So we have to approximate the set $k^f_{11,0}(t-1), \ldots k^f_{11,n-1}(t-1)$ by the set $k^f_{11,0}(t-n), \ldots k^f_{11,n-1}(t-1)$.

To summarize, the ARX lattice algorithm is obtained by iterating the errors $e^y_m(t)$, $r^y_m(t)$ and $r^u_m(t)$ up to order n, exactly as in the AR case and then applying the OBE algorithm (to each of the errors) to obtain the time updates of the forward and backward coefficients. Thus at each time step, the OBE algorithm has to be applied three times. Parameter updating at each time step depends, as before, on the information content of the data sample.

For the sake of completeness, it must be added that the ARX model considered (Eq.5) does not have a term of the form $b_0 u(t)$. This is because the predicted value of $y(t)$ is based on data up to time t-1, hence the estimate of b_0 cannot be obtained from the PARCOR coefficient matrices K and K^*, if the present lattice structure is used. We can argue that all real-world systems have a finite amount of delay between input and output and so the term $b_0 u(t)$ need not be included in the difference equation for $y(t)$. Recently, however, a lattice structure has been formulated from which the coefficient b_0 can be obtained [10].

5. SIMULATION RESULTS

Simulations have been performed for AR and ARX models. The noise sequence is assumed to be independent and identically distributed with zero mean and has a uniform distribution in $[-0.1, +0.1]$. The system models and simulation results for two different cases are presented below.

1. A fourth order autoregressive process (AR(4)) was considered. The difference

equation for the process was

$$y(t) = -0.6\, y(t-1) - 1.58\, y(t-2) - 0.464\, y(t-3) - 0.5576\, y(t-4) + v(t)$$

The mean square error (mse) of the output of the RLS and the OBE lattice is plotted in Fig.1 . The mean squared error was obtained by averaging the square of the error at every iteration over 25 runs . The plot shows that the OBE lattice algorithm converges in less than 100 time steps . However it has a larger mean square error than the RLS lattice. The asymptotic mse of the RLS lattice is close to the noise variance ($\sigma_v^2 = 0.0033$) just as in the transversal case. It has been shown [7], that the asymptotic output squared error for the transversal implementation of the OBE is constrained to lie in the interval $[0, \gamma^2]$. Since $\gamma^2 = 0.01$, the mean squared error for the OBE lattice is also expected to be in the interval $[0, 0.01]$.

2. A second order auto regressive with auxiliary input (ARX(2,2)) lattice model was simulated. The difference equation for the process was

$$y(t) = -0.4\, y(t-1) - 0.29\, y(t-2) + u(t-1) - 0.4\, u(t-2) + v(t)$$

The mean squared error of the output over 25 runs of the RLS and the OBE lattice algorithms is plotted in Fig.2 The plot again shows that the OBE lattice algorithm converges almost as fast as the RLS lattice algorithm. The mean squared error of both the algorithms is in the same range.

6. CONCLUSIONS

The lattice structure implementation of a new parameter estimation scheme has been presented. The resulting AR and ARX lattices have a fast convergence rate and low mean square error. The order of computation involved is not significantly greater than the direct implementation.

ACKNOWLEDGEMENT

This work has been supported in part by the National Science Foundation under Grant ECS- 85052/8 and in part by Naval Air Systems Command under an ONR contract N00014 - 84 -K - 0482

REFERENCES

[1] Morf, M. , A. Viera, and D. T. Lee, " Ladder forms for identification and speech processing, " *Proc. 1977 IEEE Conf. Decision and Control* , New Orleans, pp. 1074-1078

[2] Griffiths, L.J., " A continously adaptive filter implemented as a lattice structure," *Proc. 1977 Int. Conf. Acoustics Speech and Signal Process.* Hartford, Connecticut.

[3] Honig, M.L. , " Comparison of adaptive linear prediction algorithms in ADPCM, "*IEEE Trans. on Commun.* , Vol. COM-30, No. 7, July 1982, pp. 1775-1785.

[4] Huang, Y.F. , " A recursive estimation algorithm using selective updating for

spectral analysis and adaptive signal processing, " *IEEE Trans. Acoust. Speech and Signal Process.* , Vol. ASSP-34, No.5, October 1986, pp.1331-1334.

[5] Peterson, B.B. and K.S. Narendra , " Bounded Error Adaptive control," *IEEE Trans. Automatic Control* , Vol. AC-27,No.6, Dec. 1982, pp. 1161-1168.

[6] Honig, M.L. and D.L. Messerschmitt, (1984), *Adaptive Filters- Structures, Algorithms, and Applications,* Kluwer Academic Publishers.

[7] Lee, D.T. , B.Friedlander, and M.Morf, " Recursive ladder algorithms for ARMA modelling, " *Proc. 19th IEEE Conf. Decision and Control,* Albuquerque, N.M. , Dec. 1980, pp.1225-1241.

[8] Ljung L. , and T. Söderström, (1983) , *Theory and Practice of Recursive Identification,* MIT Press, Cambridge, Mass.

[9] Dasgupta S., and Y.F.Huang , "Asymptotically convergent modified recursive least-squares with data-dependent updating and forgetting factor, " *Proc. 24th IEEE Conf. Decision & Control* , Fort Lauderdale, Fl. , Dec. 1985, pp.1067-1071.

[10] Lim Y.C. , and S.R. Parker , " On the identification of systems from data measurements using ARMA lattice models ", *IEEE Trans. Acoust. Speech and Sig. Process.* , Vol. ASSP-34, No.4, Aug.1986, pp. 824-828

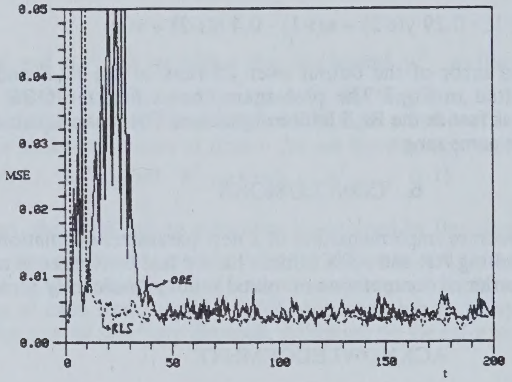


Fig. 1 Output mean squared error of the OBE and RLS lattice algorithms for an AR(4) model.

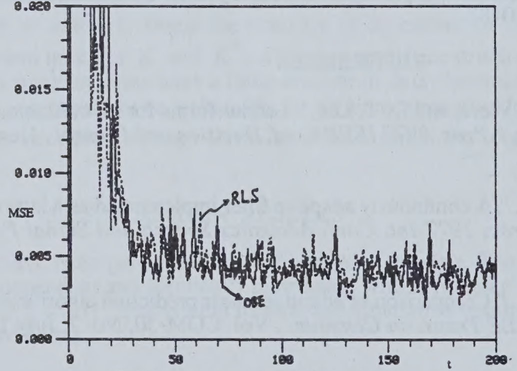


Fig. 2 Output mean squared error of the OBE and RLS lattice algorithms for an ARX(2,2) model

A NEW ESTIMATION ALGORITHM FOR SYSTEMS WITH DYNAMICAL MULTIPLICATIVE NOISE

B.S CHOW AND W.P.BIRKEMEIER
Department of Electrical and Computer Engineering
University of Wisconsin- Madison
Madison WI 53705

ABSTRACT

An optimal linear MMSE estimator, which is also recursive, is developed for a system with multiplicative noise in the measurement model. The estimated signal and the multiplicative noise are modeled by linear state variable equations. The recursiveness of the algorithm is achieved by obtaining a recursive innovation process, which is the linear combination of the current data and the previous innovation, estimate, and data. The estimator is the linear combination of the previous estimate and the current innovation rather than the conventional form which is a linear combination of the previous data and the current estimate. In addition, the off-line computation for the cofficients (named as Kalman gain for Kalman's filtering) is also recursive.

INTRODUCTION

The process of measuring a physical quantity is usually determined by the coupling between the measured object and the measuring device. The most significient coupling is a multiplicative relationship. If there is some randomness in the measureing device itself, there must be a multiplicative noise involved. Kalman's filtering [1], [2] provides a linear recursive MMSE(minimum mean suare error) estimate for signals corrupted by additive noise. In this paper, the same state-variable signal model as Kalman's is used but the measurement model has a multiplicative noise component added, and the optimal linear MMSE estimator is achieved recursively. Specifically, multiplicative noise exists in the following important cases: fading or reflection of the transmitted signal over ionospheric channel, and also certain situations involving sampling, gating, or amplitude modulation [3],[4].

Several different multiplicative noise processes which have been used in the measurement model are described briefly here. The process known as the uncertain observation, contains or does not contain signal according to a random digital switching process : 0 or 1. For the above model, Nahi [5] derived an optimal linear MMSE recursive filter to estimate the signal. Monzingo [6] extended Nahi's results to optimal linear smoothing, of which Tugnait [7] has studied the stability. Hadidi and Schwartz [8] developed,using a two-state Markov chain, a more general model for the switching sequence. They also claimed that the optimal linear MMSE filter can not be recursive for that general case. Wang [9] reached the same conclusion by a different approach. Some nonlinear estimations based upon the Bayesian approach are found in the literature [10],[11]. Little work has been done for the case of multiplicative noise with a continuous range of values. Rajasekaran [12] developed a linear MMSE recursive estimator for the continuous white noise case , and Tugnait [13] has analyzed the stability of the estimators developed by Rajasekaran.

In this paper, the continuous multiplicative noise is not required to be white and is instead described by a dynamic model. More clearly, in the measurement model, the data is the product of two dynamic random system's outputs plus an additive noise. Rajasekaran's model turns out to be a special case of our model, but his approach is not suitable for our more general model because the non-white multiplicative noise make the conventional form of the innovation process (or residual, defined in ,e.g., [14], [15])invalid. . Extended-Kalman-filtering [16] which uses linearization approximation on the nonlinear model is another possible approach to our problem, but it suffers from modeling error, and its Kalman-gain also depends upon the data and then can not be computed in advance.

The approach of this paper is to find a recursive relationship for the innovation process instead of just finding the recursive relationship of the estimator itself. For Kalman's filter and Rajasekaran's work, the innovation process is always the new data minus a linear transformation of the previous estimate. This form of innovation process leads to the usual form of a recursive estimator (for example,[6]-[10], [17]) in which the new estimate is the linear combination of the new data and the previous estimate. However, for the case of a dynamic multiplicative noise, and some other cases [8], it is impossible to achieve a solution by attempting this form. In this paper, a more general form of recursive filter is proposed, in which the new estimate is the linear combination of the previous estimate and a new term which can be obtained by another recursive relation.

PROBLEM FORMULATION

Consider the following system:

$$X(k+1) = A_k X(k) + B_k U(k) \qquad (1)$$

$$r(k+1) = c_k r(k) + d_k v(k) \qquad (2)$$

$$Z(k+1) = r(k) H_k X(k) + F_k N(k) \qquad (3)$$

where 1) $X(k)$,$r(k)$,$N(k)$,and $Z(k)$ are signal, multiplicative noise, additive noise and data respectively.

2) $U(k)$ and $v(k)$ are the generating random sequences for the signal and the multiplicative noise.

3) A_k and B_k are signal system matrices.
H_k andF_k are output matrices involved in the measurements.
c_k and d_k are scalar cofficients.

4) $X(k)$ is a $n*1$ vector, $r(k)$ is a scalar, and $Z(k)$ is a $m*1$ vector,with the other terms dimensions consistent.

the notation obeys the following two rules in this paper :

1) Random variables are distinguished from deterministic

constants by their time arguments being parenthesized instead of being subscripted.

2) Matrices and vectors are distinguished from scalars by being written in upper cases.

Next, it is assumed that

(A1) The processes U, v, and N, and the initial states X(0) and r(0) are mutually independent. also U, v, and N are all white.

(A2) x(0), r(0) ,U, v, and N are all with known means and variances.

(A3) H_k, the generalized inverse of H_k,always exists.

Under the above assumptions, a goal is to obtain $\hat{X}(k/k)$, where $\hat{X}(k/k)$ is the estimate of X(k) by the linear combination of Z(1), Z(2),...,and Z(k) such that the error of the estimate is minimized in the mean-square sense. It is derived in this paper that the optimal estimator can be achieved by a recursive algorithm.

METHODOLOGY

A linear MMSE estimation problem can be translated into a projection problem in Hilbert space. The estimated vector random variable is regarded as a vector set, and the estimating vector random variables as vector spaces. The linear MMSE estimation is then to project that vector set on a vector space. Therefore $\hat{X}(k/K)$ is the projection of X(k) on the space S(k) where S(k) is the space direct-summed by Z(1), Z(2),..., and Z(k). Let L(k)'s be the innovation process [14, 15] for Z(k)'s. Because L(k+1) is in the orthogonal compliment of the space S(k), it is clear that

$$\hat{X}(k+1/k+1) = \hat{X}(k+1/k) + \hat{X}(k+1/L(k+1)). \tag{4}$$

Because U(k) is orthogonal to (in the orthogonal compliment of) S(k), eqation (1) leads to

$$\hat{X}(k+1/k) = A_k\hat{X}(k/k). \tag{5}$$

The projection of the vector set X(k+1) onto the vector space L(k+1) can be achieved by the outer product. Then,

$$\hat{X}(k+1/L(k+1)) = E[X(k+1)L^T(k+1)]E^{-1}[L(k+1)L^T(k+1)]L(k+1).$$

Note that if $E[L(k+1)L^T(k+1)]$ is singular, the pseudo-inverse is used instead of the inverse. As a result,

$$\hat{X}(k+1/k+1) = A_k\hat{X}(k/k) + E[X(k+1)L^T(k+1)]E^{-1}[L(k+1)L^T(k+1)]L(k+1). \tag{6}$$

Referring to (6), L(k+1) determins whether $\hat{X}(k+1/k+1)$ is recursive or not. For the case of Kalman filter,

$$L(k+1) = Z(k+1)-H_{k+1}A_k\hat{X}(k/k). \tag{7}$$

Substituting (7) in (6) yields

$$\hat{X}(k+1/k+1) = Q_{1,k+1}Z(k+1) + Q_{2,k+1}\hat{X}(k/K). \tag{8}$$

where $Q_{1,k+1}$ and $Q_{2,k+1}$ are the approriate matrix. As a matter of fact, the conventional recursive estimator is assumed in the form of (8). For our problem, since the old information can not be lumped into the previous estimate, the new innovation then can not be obtained by (7). Therefore, the recursive estimator can not be achieved in the form (8). Instead, the following more general form of a recursive estimator is proposed:

$$\hat{X}(k+1/k+1) = Q_{1,k+1}\hat{X}(k/k) + Q_{2,k+1}L(k+1). \tag{9}$$

where L(k+1) can be recursively achived. For example, in our paper, L(k+1) can be recursively obtained by the linear combination of Z(k),Z(k+1), and L(k).

DERIVATION OF FUNDAMENTAL EQUATIONIONS

Part I : zero-mean case

For the system described by (1), (2), and (3), under the assumption (A1), (A2), and (A3), we further assume

(A4) The input U, v, additive noise N, and the initial states X(0), r(0) are all with zero-mean.

We first derive the recursive equation for the innovation process. Substituting (1),(2) into (3) for Z(k+1), and expanding all the terms yields

$$Z(k+1) = c_k H_{k+1} A_k r(k) X(k) + W(k) \qquad (10)$$

where

$$W(k) = F_{k+1} N(k+1) + d_k v(k) H_{k+1} A_k X(k) + c_k r(k) H_{k+1} B_k U(k) + d_k v(k) H_{k+1} B_k U(k). \qquad (11)$$

By (A3), $r(k)X(k) = \widehat{H}_k H_k r(k) X(k).$ (12)

Combining (3) and (12) yields,

$$X(k) = \widehat{H}_k Z(K) - \widehat{H}_k F_k N(k). \qquad (13)$$

Substituting (13) in (10) yields

$$Z(k+1) = c_k H_{k+1} A_k \widehat{H}_k Z(k) - c_k H_{k+1} A_k \widehat{H}_k F_k N(k) + W(k). \qquad (14)$$

Equation (14) is the basic equation in part one. Let S(k) be the linear space which is the direct-sum of the subspaces Z(1),Z(2),...,and Z(k). The innovation L(k+1) is the projection of Z(k+1) on to the orthogonal compliment of S(k). To obtain L(k+1), we note the following relationships of Z(k), N(k), and W(k) with S(k).

Z(k) is obviously in S(k).

W(k) is orthogonal to S(k) because:

From assumption (A1), v(k) is independent of any function of X(k),Z(1),Z(2),...,and Z(k). Also, v(k) is zero-mean by assumption (A4). Therefore, $E[d_k v(k) H_{k+1} A_k X(k) Z^T(i)] = 0$ for all Z(1) to Z(k). In other words, $d_k v(k) H_{k+1} A_k X(k)$ is in the orthogonal compliment of S(k). For simplicity, we say

$$d_k v(k) H_{k+1} A_k X(k) \perp S(k). \qquad (15)$$

Similarly, $c_k r(k) H_{k+1} B_k U(k) \perp S(K).$ (16)

$$d_k v(k) H_{k+1} B_k U(k) \perp S(k). \qquad (17)$$

Also, from assumption (A1) and (A4), it follows

$$F_{k+1} N(k+1) \perp S(k). \qquad (18)$$

Combining (15) , (16) , (17), and (18) yields

$$W(k) \perp S(k). \qquad (19)$$

Next, N(k) is examined:

Decompose N(k) into $N_1(k)$ and $N_2(k)$ i,e.

$$N(K) = N_1(k) + N_2(K). \qquad (20)$$

where $N_1(k)$ is in S(k), and

$N_2(k)$ is in the orthogonal compliment of S(k).

So far, it is clear that

$$L(k+1) = -c_k H_{k+1} A_k \widehat{H}_k F_k N_2(k) + W(k). \qquad (21)$$

However, (21) is still not applicable because $N_2(k)$ and W(k) are unknown.

Combining (14) ,(20) and (21), another form of L(k+1) is obtained:

$$L(k+1) = Z(k+1) - c_k H_{k+1} A_k \widehat{H}_k Z(k) - c_k H_{k+1} A_k \widehat{H}_k F_k N_1(k) \qquad (22)$$

$N_1(k)$ is still unknown in (22).
From assumption (A1) and (A4)

$$N(k) \perp S(k-1). \tag{23}$$

Because $N_2(k)$ is by definition in the orthogonal compliment of $S(k)$, it must follow that $N_2(k) \perp S(k-1)$. (24)
Since $N_1(k)=N(k)-N_2(k)$, by (23) and (24) it follows that

$$N_1(k) \perp S(k-1). \tag{25}$$

(Note that (25) can not be deducted from (23) alone)
Since $N_1(k)$ is in $S(k)$ and also in the orthgonal compliment of $S(k-1)$, $N_1(k)$ must be in the space $L(k)$. Therefore, the projection of $N_1(k)$ on $L(k)$ must yields $N_1(k)$. That is

$$N_1(k) = E[N_1(k)L(k)]E^{-1}[L(k)L^T(k)]L(k). \tag{26}$$

So, $N_1(k)$ is expressed in terms of $L(k)$. Substituting (26) in (22), the applicable form of $L(k+1)$ is achieved :

$$L(K+1)=Z(k+1) - c_k H_{k+1} A_k \hat{H}_k Z(k) + c_k H_{k+1} A_k \hat{H}_k F_k E[N_1(k)L^T(k)]E^{-1}[L(k)L^T(k)]L(k). \tag{27}$$

The inverse matrix in (27) is substituted by the pseudo-inverse if the inverse does not exist. The coefficient matrix $E[N_1(k)L^T(k)]$ can be further simplified to $E[N(k)N^T(k)]F_k$. The proof is to postmultiply (27) by $N(K+1)$ and using $E[N_1(k+1)L^T(k+1)] = E[N(k+1)L^T(k+1)]$. Therefore, the final form of (27) is

$$L(K+1) = Z(k+1) - c_k H_{k+1} A_k \tilde{H}_k Z(k) + c_k H_{k+1} A_k \tilde{H}_k F_k E[N(k)N^T(k)]F_k E^{-1}[L(k)L^T(k)]L(k). \tag{28}$$

Recall (6). The estimator is

$$\hat{X}(k+1/k+1) = A_k \hat{X}(k/k) + E[X(k+1)L(k+1)]E^{-1}[L(k+1)L^T(k+1)]L(k+1). \tag{29}$$

(28) and (29) are for on-line estimation.

The coefficient matrices in (28) and (29): $E[L(k+1)L^T(k+1)]$ and $E[X(k+1)L^T(k+1)]$ can be recursively computed as proved below and prepared for on-line estimation. (Note that $E[N(k)N(K)]$ is the known varance of the additive noise.)

In order to compute $E[L(k+1)L^T(k+1)]$, postmultiply (28) by $L(k+1)$ and take the expectation. Since $L(k+1)$ is in the orthogonal compliment of $Z(k)$ and $L(k)$ respectively, it follows that

$$E[L(k+1)L^T(k+1)] = E[Z(k+1)L^T(k+1)]. \tag{30}$$

The right side of (30) can be expanded by postmultiplying (3) by $L(k+1)$, taking the expectation and simplifing $E[N(k+1)L^T(k+1)]$ to $E[N(k+1)N^T(k+1)]$. Thus,

$$E[L(k+1)L^T(k+1)] = H_{k+1}E[r(k+1)X(k+1)L^T(k+1)] + F_{k+1}E[N(k+1)L^T(k+1)]F^T_{k+1} \tag{31}$$

Again, the first term of the right side of (31) can be expanded by premultiplying (28) by $r(k+1)X(k+1)$, taking the expectation, and using (1), (2), and (3) to simplify the terms. Then, a recursive form of $E[r(k+1)X(k+1)L^T(k+1)]$ is obtained as follows:

$$\begin{aligned} &E[r(k+1)X(k+1)L^T(k+1)] = \\ &c_k A_k E[r(k)X(k)L^T(k)]F_k E^{-1}[N(k)N^T(k)]F^T_k H^T_k A^T_k H^T_{k+1} + \\ &E[r^2(k+1)]B_k E[U(k)U^T(k)]B^T_k H^T_{k+1} + \\ &d_k^2 E[v^2(k)]A_k E[X(k)X^T(k)]A_k^T H_{k+1}^T \end{aligned} \tag{32}$$

Note that

$$E[X(k+1)X^T(k+1)] = A_k E[X(k)X^T(k)]A_K^T + B_k E[U(k)U^T(k)]B_k^T,$$
$$E[r^2(k+1)] = c_k^2 E[r^2(k)] + d_k^2 E[v^2(k)],$$

$E[U(k)U^T(k)]$ and $E[v^2(k)]$ are known variances from assumption (A2). Therefore, $E[L(k+1)L^T(k+1)]$ can be recursively computed by (31) and (32).

The computation of $E[x(k+1)L^T(k+1)]$ can be similarly obtained as follows:

$$E[X(k+1)L^T(k+1)] = c_kA_kE[X(k)L^T(k)]E^{-1}[L(k)L^T(k)]F_kE[N(k)N^T(k)]F_k{}^T\widetilde{H}_k{}^TA_k{}^TH_{k+1}{}^T \quad (33)$$

part II. Non-zero mean case

In this part, the assumption (A4) is removed. In other words, the inputs, the initial states, and the additive noise in the system described by (1), (2), and (3) are allowed to be non-zero mean. This is a more practical case. For example, given a control system, the deterministic components in inputs are usually very significant. Because of the nonlinearity in the measurement model, the work for this extention is not as straight-forward as the corresponding work for Kalman's filtering ,which has a linear measurement model. However, the signal and the multiplicative noise are both generated in linear models. Hence, the superposition principle can still be used for the signal and multiplicative noise. The signal $X(k)$ can be decomposed into a deterministic response X_k due to the deterministic components (means) of the inputs U_k and the initial states X_0, and a zero-mean random response $X_a(k)$ due to the random components of the inputs $U_a(k)$ and initial states $X_a(k)$. Likewise for the multiplicative noise. i.e.,

$$U(k) = U_k + U_a(k)$$
$$v(k) = v_k + v_a(k)$$
$$X(k) = X_k + X_a(k)$$
$$r(k) = r_k + r_a(k)$$

The deterministic responses are known from (A2) and equation (1) and (2). The zero-mean random response is then the interested signal and multiplicative noise model, but the measurement model is changed accordingly. Let Z_k be the deterministic component of $Z(k)$, and $Z_a(k)$ is the zero-mean random component of $Z(k)$. Likewise for $N(k)$. i.e.,

$$Z(k) = Z_k + Z_a(k).$$
$$N(k) = N_k + N_a(k).$$

It is obvious that $Z_k = r_kH_kX_k + N_k$.
Therefore the modified system can be written as follows:

$$X_a(k+1) = A_kX_a(k) + B_kU_a(k) \quad (34)$$
$$r_a(k+1) = c_kr_a(k) + d_kv_a(k) \quad (35)$$
$$Z_a(k) = r_a(k)H_kX_a(k) + r_kH_kX_a(k) + r_a(k)H_kX_k + F_kN_a(k) \quad (36)$$

Note that $Z_a(k)$ is computable from the known quantities $Z(k)$ and Z_k. The next step in deriving the estimator is to obtain the innovation sequence $L(1), L(2), \ldots$ and $L(k+1)$ for $Z_a(1)$, $Z_a(2), \ldots,$ and $Z_a(k+1)$. Following the similar but more tedious procedures as in part I, there is a similar equation as (14) obtained as followes :

$$Z_a(k+1) = c_kH_{k+1}A_k\widetilde{H}_kZ_a(k) + c_kv_kH_{k+1}A_kX_a(k) + c_kr_a(k)H_{k+1}B_kU_k + c_kH_{k+1}A_k\widetilde{H}_kF_kN_a(k) + W_a(k) \quad (37)$$

where $W_a(k) = F_{k+1}N_a(k+1) + d_kv_a(k)H_{k+1}A_kX_a(k) + c_kr_a(k)H_{k+1}B_kU_a(k) + d_kv_a(k)H_{k+1}B_kU_a(k) +$

$$r_{k+1}H_{k+1}B_kU_a(k) + d_kv_a(k)H_{k+1}X_{K+1}. \quad (38)$$

Note that the W(k)'s are white . Let $S_a(k)$ be the space which is direct-summed by $Z_a(1)$, $Z_a(2)$,..., and $Z_a(k)$. Again, (37) is studied to obtain the orthogonal part of $Z_a(k+1)$ to $S_a(k)$ in order to obtain the innovation L(k+1).

$Z_a(k)$ is in $S_a(k)$.

$W_a(k)$ is orthogonal to $S_a(k)$ for the same reson as described in part I.

The orthogonal part of $N_a(k)$ to $S_a(k)$ is $N_a(k)-E[N_a(k)L(k)]E^{-1}[L(k)L(k)]L(k)$, as described in part I.

Note that $\hat{X}_a(k/k)$ and $\hat{r}_a(k/k)$ are the projections of $X_a(k)$ and $r_a(k)$ on $S_a(k)$. Then, the orthogonal part of $X_a(k)$ and $r_a(k)$ to $S_a(k)$ are $X_a(k) - \hat{X}_a(k/k)$ and $r_a(k)-\hat{r}_a(k/k)$. Therefore it follows that

$$\begin{aligned} L(k+1) = {} & W(k) + c_kv_kH_{k+1}A_kX_a(k) - \hat{X}_a(k/k)) + \\ & c_k(r_a(k)-\hat{r}_a(k/k))H_{k+1}A_kB_kU_k + \qquad (39) \\ & c_kH_{k+1}A_kH_kF_k(N(k)-E[N(k)L(k)]E^{-1}[L(k)L^T(k)]L(k)) \end{aligned}$$

Combining (37) and (39), it follows that

$$\begin{aligned} L(k+1) = {} & Z_a(k+1) - c_kH_{k+1}A_k\hat{H}_kZ_a(k) - \\ & c_kv_kH_{k+1}A_kX_a(k/k) - c_kr_a(k/k)H_{k+1}B_kU_k + \\ & c_kH_{k+1}A_k\hat{H}_kF_kE[N(k)L^T(k)]E_{-1}[L(k)L^T(k)]L(k) \qquad (40) \end{aligned}$$

It is interesting to note that the random component of the multiplicative noise also needs to be estimated although the estimation of signal is the goal. All the other recursive relationships are similarly obtained as in part I. See the results in appendix. $\hat{X}_a(k+1/k+1)$ is computed by (6). Finally, the estimation is finished by adding up the deterministic term. i.e., $\hat{X}(k/k) = \hat{X}_a(k/k) + X_k$. Note that if the deterministic components of the input are zero, equation (37) has the same form as (14) and then the innovation has the same form as (28). In other words, even the initial states of the signal and multiplicative noise are not zero, the estimator's structure stay the same as part I if the inputs are zero-mean. Therefore the simple structure in part I can still applies to many practical situations. One example illutrates this is given in next section.

APPLICATION ON RADAR TRACKING

We modify a simple radar tracking problem given in Sheats [18] by introduceing a multiplicative noise in measurement model.For radar tracking, the modulated pulses are sent and then echoed back from the target with time delay and Dopplor frequency shift. The pulse delay t(k) and frequency shift f(k) have the following relation with the range R(k) and velocity V(k) of the target :

pulse delay $t(k) = 2R(k)/c(k)$, (41)

Dopplor frequency shift $f(k) = 2V(k)f_o/c(k)$. (42)

where c(k) is the pulse propagating speed in the medium and f_o is the carrier frequency.

However, there are a detection errors in detecting the received pulses. Hence the detected pulse delay t'(k) and Dopplor frequency f'(k) would be as follows:

detected pulse delay $t'(k) = t(k) + e_t(k)$ (43)

detected frequency shift $f'(k) = f(k) + e_f(k)$. (44)
where $e_t(k)$ and $e_f(k)$ are the errors.

If the turbulence of the medium is not considered, $c(k)$ is assumed to be constant c. Then, the detected range $R'(k)$, and velocity $V'(k)$ are computed as follows:

detected range $R'(k) = t'(k)c/2$ (45)
detected velocity $V'(k) = f'(k)c/(2f_o)$. (46)

Combining (41), (43), (45) yields (47).

$$R'(k) = R(k)c/c(k) + c\, e_t(K)/2 \tag{47}$$

Similarly, $V'(k) = V(k)c/c(k) + c\, e_f(k)/(2f_k)$. (48)

Note that $c/c(k)$ is the refraction index $n(k)$. Therefore a measurement model is as following:

$$\begin{bmatrix} R'(k) \\ V'(k) \end{bmatrix} = n(k) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R(k) \\ V(k) \end{bmatrix} + \begin{bmatrix} c\, e_t(k)/2 \\ c\, e_f(k)/2f_o \end{bmatrix} \tag{49}$$

Because the range and velocity are governed by the following dynamics, the signal model is :

$$\begin{bmatrix} R(k+1) \\ V(k+1) \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R(k) \\ V(k) \end{bmatrix} + \begin{bmatrix} 0 \\ T^2/2 \end{bmatrix} a(k) \tag{50}$$

where T is the time span between each pulse, and $a(k)$ is the random acceleration which is assumed white noise with zero-mean

The multiplicative noise $n(k)$, refration index of the turbelent medium is modeled as a AR time series:

$$n(k+1) = c_k n(k) + d_k v(k), \tag{51}$$

where $v(k)$ is the white noise.
Since (A1), (A2), and (A3) are satisfied, our algorithm applies

CONCLUSION

This paper presents a recursive method to find the innovation process for a new model with dynamic multiplicative noise. It is the recursiveness of the innovation process together with the recursiveness of the estimator that permitss us to derive a more general form for recursive estimation. Not only the on-line estimation is recursive, but the off-line computation of the cofficients (which are the counterparts of Kalman gain) is also recursive.

APPENDIX

This is a summary of recursive equations for non-zero mean case.
On-line computation

$$\hat{X}_a(k+1/k+1) = A_k\hat{X}_a(k/k) + E^{-1}[X_a(k+1)L^T(k+1)]L(k+1)$$

$$L(k+1) = Z_a(k+1) - c_kH_{k+1}A_k\hat{H}_kZ_a(k) + c_kv_kH_{k+1}A_k\hat{X}_a(k/k) + c_kr_a(k/k)H_{k+1}B_kU_k + c_kH_{k+1}A_k\hat{H}_kF_kE[N(k)N^T(k)]F_k{}^TE^{-1}[L(k)L^T(k)]L(k)$$

$$\hat{r}_a(k+1/k+1) = c_k\hat{r}_a(k/k) + E[r_a(k+1)L^T(k+1)]E^{-1}[L(k+1)L^T(k+1)]L(k+1)$$

Off-line computation

$$\begin{gathered}E[X_a(k+1)L^T(k+1)] = d_k v_k A_k E[X_a(k)X_T(k)]A_k^T H_{k+1}^T + \\ r_{k+1} B_k E[U_a(k)U_a^T(k)]B_k^T H_{k+1}^T + c_k v_k A_k E[X_a(k)\hat{X}_a(k/k)]A_{kTHk+1}^T + \\ c_k A_k E[X_a(k)\hat{r}_a(k/k)]U_k^T B_k^T H_{k+1}^T + \\ c_k A_k E[X_a(k)L^T(k)]E^{-1}[L(k)L^T(k)]F_k E[N(k)N^T(k)]F_k^T H_k^T A_k^T H_{k+1}^T\end{gathered}$$

$$\begin{gathered}E[r_a(k+1)L^T(k+1)] = c_k 2E[r_a^2(k)]U_k^2 B_k^T H_{k+1}^T + \\ d_k^2 E[v_a^2(k)]X_{k+1}^T H_{k+1}^T + c_k^2 v_k E[r_a(k)\hat{X}_a(k/k)]A_{kTHk+1}^T + \\ c_k^2 E[r_a(k)\hat{r}_a(k/k)]U_k^T B_k^T H_{k+1}^T + \\ c_k^2 E[r_a(k)L^T(k)]E^{-1}[L(k)L^T(k)]F_k E[N(k)N^T(k)]F_k^T H_k^T A_k^T H_{k+1}^T\end{gathered}$$

$$\begin{gathered}E[r_a(k+1)X_a(k+1)L^T(k+1)] = d_k^2 E[v_a^2(k)]A_k E[X_a(k)X_a^T(k)]A_k H_{k+1} + \\ E[r_a^2(k+1)]B_k^T E[U_a(k)U_a^T(K)]B_k^T H_{k+1}^T + \\ c_k^2 v^k A_k E[r_a(k)X_a(k)\hat{X}_a^T(k/k)]A_k^T H_{k+1}^T + \\ c_k^2 A_k E[r_a(k)X_a(k)\hat{r}_a(k/k)]U_k^T B_k^T H_{k+1}^T + \\ c_k A_k E[r_a(k)X_a(k)L^T(k)]E^{-1}[L(k)L^T(k)]F_k E[N(k)N^T(k)]F_k^T H_k^T A_k^T H_{k+1}^T\end{gathered}$$

$$\begin{gathered}E[L(k+1)L^T(k+1)] = F_{k+1} E[N(k+1)N^T(k+1)]F_{k+1}^T + \\ H_{k+1} E[r_a(k+1)X_a(k+1)L^T(k+1)] + H_{k+1} r_{k+1} E[X_a(k+1)L^T(k+1)] + \\ H_{k+1} X_{k+1} E[r_a(k+1)L^T(k+1)]\end{gathered}$$

$$\begin{gathered}E[r_a(k+1)X_a(k+1)X_a(k+1/K+1) = c_k A_k E[r_a(k)X_a(k)\hat{X}_a^T(k/k)A_k^T + \\ E[r_a(k+1)X_a(k+1)L^T(k+1)]E^{-1}[L(k+1)L^T(K+1)]E[X_a(k+1)L^T(k+1)]\end{gathered}$$

$$\begin{gathered}E[r_a(k+1)X_a(k+1)\hat{r}_a(k+1/K+1) = c_k^2 A_k E[r_a(k)X_a(k)\hat{r}_a(k/k)] + \\ E[r_a(k+1)X_a(k+1)L^T(k+1)]E^{-1}[L(k+1)L^T(K+1)]E[r_a(k+1)L^T(k+1)]\end{gathered}$$

$$\begin{gathered}E[X_a(k+1)\hat{X}_a(k+1/k+1)] = A_k E[X_a(k)\hat{X}_a(k/k)]A_k^T + \\ E[X_a(k+1)L^T(k+1)]E^{-1}[L(k+1)L^T(k+1)]E[X_a(k+1)L^T(K+1)]\end{gathered}$$

$$\begin{gathered}E[X_a(k+1)\hat{r}_a(k+1/k+1)] = c_k A_k E[X_a(k)\hat{r}_a(k/k)] + \\ E[X_a(k+1)L^T(k+1)]E^{-1}[L(k+1)L^T(k+1)]E[r_a(k+1)L^T(K+1)]\end{gathered}$$

$$\begin{gathered}E[r_a(k+1)\hat{X}_a(k+1/k+1)] = c_k E[r_a(k)\hat{X}_a(k/k)]A_k^T + \\ E[r_a(k+1)L^T(k+1)]E^{-1}[L(k+1)L^T(k+1)]E[X_a(k+1)L^T(K+1)]\end{gathered}$$

$$\begin{gathered}E[r_a(k+1)\hat{r}_a(k+1/k+1)] = c_k^2 E[r_a(k)\hat{r}_a(k/k)] + \\ E^{-1}[r_a(k+1)L^T(k+1)]E[r_a(k+1)L^T(K+1)]\end{gathered}$$

ACKNOWLEDGEMENT

The disscussion with professor C. De Marco and T. Kurtz helps very much for writting this paper.

REFERENCES

[1] R. E. Kalman, "A new approach to linear filtering and linear filtering and prediction problems," Trans. ASME, J. Basic Eng., ser. D, vol. pp. 35-45, Mar. 1960.

[2] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," Trans. ASME, J. Basic Eng., ser. D, vol. 83,pp. 95-108,Mar. 1961.

[3] S. G. Loo, "Statistical design of optimum multivariable systems whose inputs are corrupted by multiplicative noise," IEEE Trans. Automat. Contr. vol. AC-13, pp. 300-301, June 1968.

[4] H. R. Ramer, Statistical Communication Theory and Applications: Englewood Cliffs, N. J.: Prentice-Hall, 1969

[5] N. E. Nahi, "Optimal recursive estimation with uncertain observation," IEEE trans. Inform. Theory, vol. IT-15, pp.457-462, July,1969.

[6] R. A. Monzingo, "Discrete optimal linear smoothing for systems with uncertain observation," IEEE Trans. Inform. Theory, vol. IT-21, pp.271-275, may 1975

[7] J, K. Tugnait, "Asymptotic stability of the MMSE linear filter for systems with uncertain observations", IEEE Trans. Automat. Contr., vol. IT-21, pp247-250, Mar. 1981.

[8] M. T. Hadidi and S. C. Schwartz, "Linear recursive state estimator under uncertain observations," IEEE Trans. Automat. Contr., vol. AC_24, No. 6 pp. 944-948, Dec. 1979

[9] X. Wang, "Recursive algorithm for linear LMSE estimator under uncertain observations," IEEE Trans. Automat. Contr., vol. AC-29, pp. 853-854, Sep. 1984

[10] H. Derin, "Optimal sequential smoothing with uncertain observation", 22'th Annual Allerton Confer. pp 613-621.

[11] A. G Jaffer and S. C. Gupta, "Recursive Bayesian estimation with uncertain observations", IEEE Trans. Inform. Theory, vol. IT-17, pp614-616, Sep. 1984.

[12] P. K. Rajasekaran and N. Satyanarayana and M. D. Srinath,"Optimum linear estimation of stochastical signals in the presence of multiplicative noise," IEEE Trans. Aerosp. Electron. Syst. vol. AES-7, pp. 462-468, May 1971.

[13] J. K. Tugnait, "Stability of optimum linear estimators of stochastical signals in white multiplicative noise," IEEE Trans. Automat. Contr., vol. AC-26, pp. 757-761, June 1981.

[14] T. Kailath, "An innovations approach to least-squares estimation. part I: Linear filtering in additive white noise," IEEE trans. Automat. Contr. vol. AC-13 pp. 646-655. Dec. 1968

[15] R. W. Sorenson, "Least-squares estimation: from Gauss to Kalman," IEEE Spectrom, vol. 7, pp.63-68,July 1970. PP. 146-181,1974

[16] R. E. Kopp and R. J. Oxford, "LInear regression applied to system identification and adaptive control system, " AIAA

[17] Gelb, Applied Optimal Estimation: The Analytic Science poration, Mass: the M.I.T. press, 1974

[18] L. Sheats, " The Kalman filter," : Radar Technolog edited by E. Brookner, Raytheon Company, Artech House Inc. pp.373-pp.378.

FREQUENCY DOMAIN CONDITIONS FOR DISTURBANCE DECOUPLING BY OUTPUT FEEDBACK

R. P. SINGH
Bell Communications Research
331 Newman Springs Road, NVC-2Z-325
Red Bank, NJ 07701-7020

K. S. NARENDRA
Department of Electrical Engineering
Yale University
New Haven, CT 06520-2157

Abstract - Given a plant description in terms of its transfer matrices, we derive necessary and sufficient conditions in the frequency domain for the solvability of the multivariable disturbance decoupling problem with independent controlled and measured outputs.

I. INTRODUCTION

We consider the problem of decoupling the effects of disturbances at the controlled output ports using a dynamic feedback compensator from the measured outputs. Reference [1] has derived the geometric solvability conditions in state space using subspace arithmetic. The set of geometric conditions has been translated into frequency domain conditions in [2] using a translation rule from subspaces in the state space to the corresponding degree conditions for the related elements of transfer matrices. It has been mentioned in [2] that the frequency domain conditions generically hold in the parameter space of the coefficients of the transfer functions.

In this paper, we directly derive general frequency domain conditions for the solvability of the disturbance decoupling problem by output feedback and show that the simple conditions given in [2] are special cases of these general conditions. Furthermore, the solvability of the problem is shown to be closely related to the problem of decoupling of a plant by a state feedback and a nonsingular feedforward gain controller and its dual form. We also argue that if the initial description of the plant is given in terms of its transfer matrices, these simple conditions are not generic since these are not continuous with respect to their arguments.

Section II contains a precise statement of problem. Section III discusses some preliminary results needed for the derivation of the main results in Section IV. Section V provides some concluding remarks.

II. STATEMENT OF PROBLEM

Consider the plant shown in Figure 1. The plant has two input vectors, the control input vector $u(t) \in R^m$ and the disturbance input vector $d(t) \in R^r$, and the two output vectors, the controlled output vector $y(t) \in R^p$ and the measured output vector $z(t) \in R^q$. The vectors $y(t)$ and $z(t)$ may have common components. The plant is assumed to be linear, time-invariant, finite-dimensional and can be represented in terms of its transfer matrices from input vectors to output vectors as

$$\begin{pmatrix} y(t) \\ z(t) \end{pmatrix} = \begin{pmatrix} W_1(s) & W_2(s) \\ W_3(s) & W_4(s) \end{pmatrix} \begin{pmatrix} u(t) \\ d(t) \end{pmatrix} \tag{2.1}$$

The transfer matrices $W_i(s)$, $i = 1$ to 4 are of consistent dimensions and are strictly proper, i.e., the elements of each of the transfer matrices belong to the set of proper rational transfer functions, denoted by $R_p(s)$*. The controller is a linear, time-invariant, finite dimensional dynamical system with transfer matrix $K(s)$ which relates the measured output vector to the control vector, i.e.,

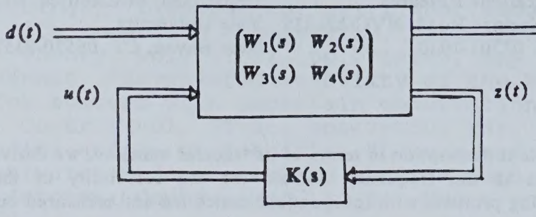


Figure 1. The Plant and the Controller

$$u(t) = -K(s)\,z(t) \tag{2.2}$$

Given $W_i(s)$, $i = 1$ to 4 with elements in $R_p(s)$, the disturbance decoupling problem by output feedback is to find a causal transfer matrix $K(s)$ such that the closed-loop transfer matrix from the disturbance input vector $d(t)$ to the controlled output vector $y(t)$ is identically zero. The closed-loop transfer matrix can be computed from (2.1) and (2.2), and thus the problem requires that

$$-W_1(s)\,K(s)\,(I + W_3(s)\,K(s))^{-1}\,W_4(s) + W_2(s) \equiv 0 \tag{2.3}$$

III. PRELIMINARIES

The mechanism by which (2.3) is satisfied is that $z(t)$ provides an estimate of $d(t)$ through the dynamical system $W_4(s)$, and the control vector $u(t)$ uses this information via the controller $K(s)$ to "drive" $y(t)$ to zero. If every component of $y(t)$ were to go to zero independently, then it is necessary that the plant be output function controllable. This, according to the results in [3], implies that the rank of $W_1(s)$ be p ; i.e., the dimension of the control vector should be more than or equal to that of the controlled output vector. Hence, $m \geq p$. For the sake of simplicity, it will be assumed that $m = p$. (Those extra control input components can be ignored whose corresponding columns are linearly dependent on the other p columns of the transfer matrix $W_1(s)$).

The dual of output function controllability is the input function reconstructibility. This applies to the problem of estimating $d(t)$ through $z(t)$. For each element of $d(t)$ to be independently "reconstructed" via the vector $z(t)$, the rank of $W_4(s)$ should be r. That is, $q \geq r$. Again, for simplicity, $q = r$ is assumed. Thus, the transfer matrices $W_1(s)$ and $W_4(s)$ are square of dimensions mxm and rxr, respectively. Furthermore, it is also assumed that they are nonsingular. For if they were not, the problem can be reduced to a lower dimension problem by deleting the dependent rows and columns.

---

* With respect to a transfer function $t(s) \in R_p(s)$, the terms causal, proper (strictly proper) and differentiator-free mean the same, i.e., $\lim_{s \to \infty} t(s) =$ a nonzero constant (= zero).

Important to the development of the results in this paper is the concept of the right and left Hermite normal forms of a square, nonsingular matrix defined over a Euclidean domain, and, in particular, over $R_p(s)$ which has been shown to be as such in [4,5].

The right Hermite normal form of a square, nonsingular rational transfer matrix $T(s)$ in $R_p^{mxm}(s)$ ($R_p^{mxm}(s)$ is a set of mxm matrices with elements in $R_p(s)$) is a special lower triangular matrix in $R_p^{mxm}(s)$ obtained by performing elementary column operations on $T(s)$ as described in [4,5]. These operations are equivalent to multiplying the matrix $T(s)$ on the right with an appropriate unimodular matrix* $U(s)$ *is* $R_p^{mxm}(s)$. The right Hermite normal form of $T(s)$ is given by

$$H(s) = \begin{bmatrix} \frac{1}{\pi^{n_1}(s)} & 0 & \cdot\cdot & 0 \\ h_{21}(s) & \frac{1}{\pi^{n_2}(s)} & \cdot\cdot & 0 \\ \cdot & \cdot & \cdot\cdot & \cdot \\ \cdot & \cdot & \cdot\cdot & \cdot \\ h_{m1}(s) & h_{m2}(s) & \cdot\cdot & \frac{1}{\pi^{n_m}(s)} \end{bmatrix} \tag{3.1}$$

$$\text{where } h_{ij}(s) = \left[\frac{q_{ij}(s)}{\pi^{n_{ij}}(s)},\, n_{ij} < n_i\right], \frac{q_{ij}(s)}{\pi^{n_{ij}}(s)} \in R_p(s)$$

and n_i and n_{ij} are positive integers for $i = 2,..., m$ and $j = 1,..., m-1$. $H(s)$ is unique up to a choice of $\pi(s)$, where $\pi(s)$ is an arbitrary monic polynomial of degree one. It contains information about inherent integrators in the dynamical system T(s).

Thus,

$$T(s)\, U(s) = H(s) \tag{3.2}$$

It is known [4,6] that in equation (3.2) the compensator matrix $U(s)$ is a unimodular matrix in $R_p^{mxm}(s)$ if and only if it is equivalent to a state feedback and a constant nonsingular feedforward gain controller.

There is a class of transfer matrices in $R_p^{mxm}(s)$ whose right Hermite normal form is diagonal in structure. The equation (3.2) implies that a transfer matrix in this class can be decoupled by a state feedback and a constant nonsingular feedforward gain controller. The condition under which a transfer matrix in $R_p^{mxm}(s)$ can be decoupled by such a controller is well known [7] and is stated below. We need the following definitions.

Let $T(s)$ be a nonsingular transfer matrix in $R_p^{mxm}(s)$. Let $r_i(T(s))$ denote the minimum relative degree in the ith row of $T(s)$; i.e., $r_i \triangleq \min_j$ (degree of denominator polynomial

---

[*] A matrix $U(s)$ in $R_p^{mxm}(s)$ is unimodular if and only if its determinant is a rational function in $R_p(s)$ with zero relative degree, defined as the degree of denominator minus the degree of the numerator of the rational function. Therefore, if U(s) is unimodular, then $U^{-1}(s) \in R_p^{mxm}(s)$.

minus degree of numerator polynomial of *ijth* element of $T(s)$). Let $1xm$ constant row vector E_i be defined as

$$E_i\,(T(s)) \triangleq \lim_{s \to \infty} s^{r_i}\, T(s)$$

Lemma 3.1 [7]: Let $T(s)$ and E_i, $i=1,\ldots,m$ be as above. $T(s)$ can be decoupled by a state feedback and a constant nonsingular feedforward gain controller if and only if the constant matrix

$$E(T(s)) = \begin{bmatrix} E_1 \\ E_2 \\ \cdot \\ \cdot \\ \cdot \\ E_m \end{bmatrix} \tag{3.3}$$

is nonsingular.

We can state the following result:

Lemma 3.2: Given a nonsingular matrix $T(s)$, its right Hermite normal form $H(s)$ is diagonal in structure if and only if the constant matrix $E(T(s))$ is nonsingular.

If $E(T(s))$ is nonsingular, then the *ith* diagonal element of $H(s)$ is simply $\dfrac{1}{\pi^{r_i}(s)}$, where r_i is defined above.

Since the condition of the above lemma is in terms of the nonsingularity of a matrix defined over R, one may be tempted to conclude its genericity, and hence that the right Hermite form is generically diagonal in structure. But it is important to note that $E(T(s))$ is a function of two sets of arguments, the relative degrees r_i's and the scalar entries in E_i's. Although it is continuous with respect to the latter, it is only a discrete function of the former. Furthermore, the zero entries in $E(T(s))$ are fixed and the nonzero entries are not allowed to continuously go through zero. Hence, the genericity argument fails and we cannot conclude that $H(s)$ is generically diagonal in structure.

There is another equivalent necessary and sufficient condition in geometric terms for decoupling by a state feedback and a feedforward gain controller using the state space representation for the plant $T(s)$, given by

$$\dot{x}(t) = A\; x(t) + B\; u(t) \tag{3.4a}$$

$$y(t) = C\; x(t) \tag{3.4b}$$

where $x(t)$ is an n-tuple state vector, $u(t)$ and $y(t)$ are m-tuple input and output vectors, respectively. C, A, B are constant matrices of consistent dimensions. The control law is of the form

$$u(t) = F\; x(t) + G\; v(t) \tag{3.5}$$

where F and G are the feedback and feedforward gains of appropriate dimensions, respectively, and $v(t)$ is an m-tuple external input vector. For fixed data points for both (C, A, B) and $T(s)$, the equivalence of these conditions for decoupling has been shown by Morse and Wonham in [9]. Warren and Mitter in [10] have used the tools of algebraic geometry to show that, for a (C, A, B) representation of the plant, the geometric condition for decoupling, given by Morse and Wonham [9], is generically true; i.e., for all parameter values of (C, A, B), such a plant can almost always be decoupled by the control law (3.5). The proof given in [10] is quite involved. We present here a simpler proof for generic decoupling by the control law (3.5) in the (C, A, B) parameter space. First, we derive a sufficient condition for decoupling by the control input (3.5) in terms of (C, A, B) and, later, we demonstrate the genericity of the condition.

Proposition 3.1: If the plant has a state space representation given in (3.4), then a sufficient condition for the decoupling of the plant by the control law (3.5) is that the constant matrix CB be nonsingular. Hence, the plant (3.4) can be generically decoupled by the control law (3.5).

Proof: Let the characteristic equation of the matrix A be given by $a(s) \triangleq det[(sI-A)] = s^n + a_{n-1}s^{n-1} + \dots + a_0$. From the results of [pp. 84-85, 11], it is known that

$$H(s) \triangleq C(sI - A)^{-1} B = \frac{1}{a(s)} (r_{n-1}(s)CB + r_{n-2}(s)CAB + \dots + r_0(s)CA^{n-1}B)$$

where

$$r_i(s) = s^i + a_{n-1}s^{i-1} + a_{n-2}s^{i-2} + \dots + a_{n-i},\ (i = 1,\dots,n-1) \text{ and } r_0(s) = 1.$$

If the control law (3.5) is used, the closed-loop transfer matrix, denoted by $H(s; F,G)$, is given by

$$H(s; F,G) = C(sI - A - BF)^{-1}BG = \frac{1}{a_f(s)} (q_{n-1}(s)CB + q_{n-2}(s)C(A + BF)B + \dots$$

$$\dots + q_0(s)C(A + BF)^{n-1}B)G$$

where $a_f(s)$ is the characteristic equation of $A + BF$ and $q_i(s)'s$ are defined analogous to $r_i(s)'s$ with the coefficients obtained from $a_f(s)$. If CB is nonsingular, then with the choice of $F = -(CB)^{-1} CA$ and $G = (CB)^{-1}$, we have

$$H(s; F,G) = \frac{q_{n-1}(s)}{a_f(s)} I$$

which is a decoupled transfer matrix.

The second result now follows from the fact that $det[CB]$ is a continuous function of the elements of the matrix CB and it is singular only on the variety $det[CB] = 0$.

It has been possible to show the genericity of the conditions (either the geometric condition of [9] or the nonsingularity of the matrix CB) for decoupling because these are continuous with respect to the parameters of (C, A, B). However, the genericity of the condition in the (C, A, B) parameter space cannot be extended to the equivalent condition, namely, the nonsingularity of the matrix $E(T(s))$, in the parameter space of the equivalent transfer matrix $T(s)$. This is because the mapping from the parameter space of (C, A, B) to the parameter space of $T(s) = C(sI - A)^{-1}B$, the equivalent transfer matrix, is not bi-continuous in natural topologies, and hence the genericity of a property in the (C, A, B)

space may not be so in the parameter space of $T(s)$.

Remark: The dual forms of the definitions and other results are obtained by applying them to the transposed matrices. Analogous to $H(s)$ in (3.1), every nonsingular matrix in $R_p^{mxm}(s)$ has a left Hermite normal form whose structure is similar to $H^T(s)$. Likewise, the left Hermite form is diagonal in structure if and only if $E(T^T(s)) \triangleq F(T(s))$ is nonsingular.

For three matrices $X(s)$, $Y(s)$ and $Z(s)$ *in* $R_p^{mxm}(s)$, if $X(s) = Y(s)\,Z(s)$ holds, $Z(s)$ $(Y(s))$ is said to right (left) divide $X(s)$.

IV. MAIN RESULTS

Now, we derive conditions on different transfer matrices $W_i(s)$, $i = 1,\dots,4$ in (2.1) for a causal controller $K(s)$ to exist such that (2.3) is satisfied. After simplification of (2.3), it can be shown that ($W_i(s)$'s are nonsingular)

$$K(s) = (W_1(s) - W_2(s)\,W_4^{-1}(s)\,W_3(s))^{-1}\,W_2(s)\,W_4^{-1}(s) \tag{4.1a}$$

which can also be written as

$$K(s) = -W_1^{-1}(s)\,W_2(s)\,(W_3(s)\,W_1^{-1}(s)\,W_2(s) - W_4(s))^{-1} \tag{4.1b}$$

The terms on the right hand side of (4.1) can be expressed as (the argument 's' is suppressed for brevity):

$$(W_1 - W_2\,W_4^{-1}\,W_3)^{-1}\,W_2\,W_4^{-1} = (I - W_1^{-1}\,W_2\,W_4^{-1}\,W_3)^{-1}\,W_1^{-1}\,W_2\,W_4^{-1} \tag{4.2a}$$

$$W_1^{-1}\,W_2\,(W_3\,W_1^{-1}\,W_2 - W_4)^{-1} = W_1^{-1}\,W_2\,W_4^{-1}\,(I - W_3\,W_1^{-1}\,W_2\,W_4^{-1})^{-1} \tag{4.2b}$$

From above it is evident that causality or properness of $K(s)$ depends upon that of $W_1^{-1}(s)\,W_2(s)\,W_4^{-1}(s)$ and that the causality of one implies and is implied by the causality of the other. Since if $W_1^{-1}(s)\,W_2(s)\,W_4^{-1}(s)$ is proper, then $(I - W_3(s)\,W_1^{-1}(s)\,W_2(s)\,W_4^{-1}(s))$ and $(I - W_1^{-1}(s)\,W_2(s)\,W_4^{-1}(s)\,W_3(s))$ are unimodular matrices in $R_p^{mxm}(s)$, assuming that they are nonsingular. Nonsingularity of these matrices implies that $K(s)$ is nondegenerate. In the scalar case, when $m = r = 1$, this nonsingularity requirement implies that the two scalar outputs $y(t)$ and $z(t)$ are independent time functions. This can be summarized in the following lemma.

Lemma 4.1: The controller K(s) is causal if and only if $W_1^{-1}(s)W_2(s)W_4^{-1}(s)$ is causal.

Since the Hermite forms contain information about inherent integrators (relative degrees) in a rational transfer matrix, the causality condition can also be stated in terms of the left and right Hermite forms of the appropriate transfer matrices. Let $H_1(s)$ and $H_{2r}(s)$ be the right Hermite forms of $W_1(s)$ and $W_2(s)$, respectively. Let $H_4(s)$ and $H_{2l}(s)$ denote the left Hermite forms of $W_4(s)$ and $W_2(s)$, respectively. We have the following lemma which gives a necessary condition for the causality of $K(s)$.

Lemma 4.2: The controller $K(s)$ is causal only if (a) the right Hermite form $H_1(s)$ *of* $W_1(s)$ left divides the right Hermite form $H_{2r}(s)$ *of* $W_2(s)$, and (b) the left Hermite form $H_4(s)$ *of* $W_4(s)$ right divides the left Hermite form $H_{2l}(s)$ *of* $W_2(s)$.

Proof: From the definitions of the right and left Hermite forms of a rational matrix, it follows that

$$W_1(s)Q_1(s) = H_1(s), \quad W_2(s)Q_{2r}(s) = H_{2r}(s) \tag{4.3a}$$

$$Q_4(s)W_4(s) = H_4(s), \quad Q_{2l}(s)W_2(s) = H_{2l}(s) \tag{4.3b}$$

for some unimodular matrices $Q_1(s), Q_{2r}(s), Q_4(s)$ and $Q_{2l}(s)$, respectively, in $R_p^{mxm}(s), R_p^{rxr}(s), R_p^{mxm}(s)$ and $R_p^{rxr}(s)$.

If $W_1^{-1}(s)\ W_2(s)W_4^{-1}(s)$ is causal, then it is necessary that $W_1^{-1}(s)W_2(s)$ and $W_2(s)W_4^{-1}(s)$ be causal. Both (a) and (b) now follow by applying the above definitions to the transfer matrices $W_1^{-1}(s)\ W_2(s)$ and $W_2(s)W_4^{-1}(s)$, respectively, and using the definitions of left and right divisibility mentioned in Section III.

It is possible to state a necessary and sufficient condition, too, but the condition is not as transparent. Since by Lemma 4.2, $W_1^{-1}(s)\ W_2(s)$ is causal and is $\epsilon\ R_p^{mxr}(s)$, let its left Hermite form be $H_{12}(s)$. Similarly, by Lemma 4.2 $W_2(s)\ W_4^{-1}(s)$ is causal and is in $R_p^{mxr}(s)$, let its right Hermite form be $H_{24}(s)$.

Assuming that $K(s)$ is nondegenerate, the following lemma can be stated.

Lemma 4.3: The disturbance decoupling problem for the plant (2.1), with independent $y(t)$ and $z(t)$ vector-valued functions, is solvable using a causal controller $K(s)$ if and only if either (a) $H_4(s)$ right divides $H_{12}(s)$, or (b) $H_1(s)$ left divides $H_{24}(s)$.

Proof: The obvious proof is omitted.

The necessary and sufficient conditions in the above lemma is not very intuitive because often the Hermite forms will have to be determined by performing tedious row and column operations. However, it is possible to derive a simpler condition for solvability with some additional constraints on $W_1(s)$ and $W_4(s)$. This is more useful since the condition is easy to check. We need the following definitions.

Let r_{ij} denote the relative degree of the *ijth* element $w_{2ij}(s)$ (if $w_{2ij}(s) \equiv 0$, $r_{ij} = \infty$ is assumed) of $W_2(s)$, and q_i and p_j denote the minimum of relative degrees of the elements in the *ith* row $w_{1i}(s)$ of $W_1(s)$and the *jth* column $w_{4j}(s)$ of $W_4(s)$, respectively; i.e., $\lim_{s\to\infty} s^{r_{ij}}\ w_{2ij}(s)$ is a scalar, including zero when $w_{2ij}(s) \equiv 0$, $\lim_{s\to\infty} s^{q_i}\ w_{1i}(s)$ is a nonzero row vector in R^{1xm} and $\lim_{s\to\infty} w_{4j}(s)s^{p_j}$ is a nonzero column vector in R^{rx1}.

Lemma 4.4: If $K(s)$ is causal, then

$$r_{ij} \geq q_i + p_j, \qquad (i = 1,\ldots,m) \text{ and } (j = 1,\ldots r) \tag{4.4}$$

Proof: It is known from Lemma 4.1 that $K(s)$ is causal if and only if $W_1^{-1}(s)\ W_2(s)\ W_4^{-1}(s)$ is causal. This means that there exists a matrix $T(s)\ \epsilon\ R_p^{mxr}(s)$ such that

$$W_2(s) = W_1(s)\ T(s)\ W_4(s)$$

The *ijth* element $w_{2ij}(s)$ *of* $W_2(s)$ is then given by

$$w_{2ij}(s) = w_{1i}(s)\ T(s)\ w_{4j}(s)$$

Since $T(s)$ is a causal matrix, the *ijth* element of $W_2(s)$ must have relative degree greater than or equal to the sum of the minimum relative degree in the *ith* row of $W_1(s)$ and the *jth* column of $W_4(s)$. The result follows.

The usefulness of the above necessary condition is rather limited. The sufficiency version of the lemma is true under the condition of nonsingularity of the matrices $E(W_1(s))$ and $F(W_4(s))$. By Lemma 3.2, it is known that if and only if $E(W_1(s))$ and $F(W_4(s))$ are nonsingular, $H_1(s)$ and $H_4(s)$ are diagonal, with their *ith* and *jth* diagonal elements as $1/\pi^{q_i}(s)$ and $1/\pi^{p_j}(s)$, respectively. We use this to prove the sufficiency part in the following lemma.

Lemma 4.5: If the constant matrices $E(W_1(s)) \in R^{mxm}$ and $F(W_4(s)) \in R^{rxr}$ are nonsingular, then $W_1^{-1}(s)\, W_2(s)\, W_4^{-1}(s)$, or equivalently $K(s)$, is causal if and only if (4.4) is true.

Proof: Necessity: It follows by appealing to Lemma 4.4 above. If $W_1^{-1}(s)\, W_2(s)\, W_4^{-1}(s)$ is causal, then (4.4) is true regardless of whether $E(\cdot)$ or $F(\cdot)$ is nonsingular.

Sufficiency: To show sufficiency, we use the above stated fact regarding the diagonality of $H_1(s)$ and $H_4(s)$.

$$W_1^{-1}(s)\, W_2(s)\, W_4^{-1}(s) = Q_1(s)\, H_1^{-1}(s)\, W_2(s)\, H_4^{-1}(s)\, Q_4(s)$$

Consider the matrix $H_1^{-1}(s)\, W_2(s)\, H_4^{-1}(s)$ with $H_1(s)$ and $H_4(s)$ diagonal. It is obvious that if (4.4) is true, then the transfer matrix $H_1^{-1}(s)\, W_2(s)\, H_4^{-1}(s)$ is causal. Further, since $Q_1(s)$ and $Q_4(s)$ are unimodular in $R_p^{mxm}(s)$ and $R_p^{rxr}(s)$, respectively, causality of $W_1^{-1}(s)\, W_2(s)\, W_4^{-1}(s)$ follows.

The matrix $E(\cdot)$ (or $F(\cdot)$) is not generically nonsingular because of its discrete functional dependence on the relative degrees q_i's (p_j's). Furthermore, the zero entries are fixed and the nonzero entries are not allowed to continuously go through zero. The following examples elucidate these points. Consider two transfer matrices with $E(\cdot)$ matrices given as:

$$T_1(s) = \begin{bmatrix} \dfrac{k_1}{s+b_1} & \dfrac{k_2}{s+b_2} \\ \dfrac{k_3}{s+b_3} & \dfrac{k_4}{s+b_4} \end{bmatrix} \qquad E(T_1(s)) = \begin{bmatrix} k_1 & k_2 \\ k_3 & k_4 \end{bmatrix}$$

and

$$T_2(s) = \begin{bmatrix} \dfrac{k_1}{s+b_1} & \dfrac{k_2}{(s+b_2)^4} \\ \dfrac{k_3}{s+b_3} & \dfrac{k_4}{(s+b_4)^2} \end{bmatrix} \qquad E(T_2(s)) = \begin{bmatrix} k_1 & 0 \\ k_3 & 0 \end{bmatrix}$$

The matrix $E(T_1(s))$ is generically nonsingular in the parameter space $R^4\backslash\{0\}$. The matrix $E(T_2(s))$, however, is always singular regardless of how much parameter perturbations are allowed on k_1 and k_3, unless the relative degrees of $T_2(s)$ are permitted to change discretely.

Now, according to the discussion in Section III, if the initial data points are given in terms of (C, D, A, B, E), then the Hermite normal forms of the related transfer matrices, $W_1(s) = C(sI - A)^{-1}B$ and $W_4(s) = D(sI-A)^{-1}E$, will be generically diagonal. The result of Lemma 4.5 is generically true in this case. If, however, the initial data points are given in terms of transfer matrices $W_i(s)$, $i = 1,\ldots,4$, which is usually the case, the result of Lemma 4.5 cannot be concluded to be generic due to the lack of generic nonsingularity of $E(W_1(s))$ and $F(W_4(s))$ as discussed above.

The result (4.4) is useful, because the solvability of the disturbance decoupling problem can be directly determined by the relative degree relationship rather than by tediously computing either $W_1^{-1}(s)\, W_2(s)\, W_4^{-1}(s)$ or the related Hermite normal forms as in Lemma 4.3, although (4.4) is true for a restricted class of plants. But this result becomes even more useful in the adaptive control situation where the parameters of $W_i(s)'s$ are not known, and thus the latter quantities cannot be computed [6,8]. In this case, with the knowledge of the relative degree of each element of transfer matrices, $W_1(s)$, $W_2(s)$ and $W_4(s)$, the condition (4.4) can easily be checked. Furthermore, in adaptive control, it is know [6,8] that in case $E(W_1(s))$ $(F(W_4(s)))$ is singular, with the knowledge of relative degree of each element of $W_1(s)$ $(W_4(s))$, a dynamic pre-(post-) compensator $W_c(s)$ $(W_e(s))$ can be designed such that $E(W_1(s)\, W_c(s))$ $(F(W_e(s) W_4(s)))$ is generically nonsingular in the parameter space $R^s \backslash \{0\}$, where s is the number of nonzero entries in the matrix $E(\cdot)$ $(F(\cdot))$. In this case Lemma 4.5 can be used with transfer matrices $W_1(s)\, W_c(s)$ and $W_e(s)\, W_4(s)$. Their right and left Hermite forms, respectively, are diagonal in structure. However, use of $W_c(s)$ and $W_e(s)$ increases q_i and p_j, respectively. Therefore, (4.4) is true for a limited class of plants. Hence, there is a trade-off between the required apriori knowledge of the structure of the plant transfer matrix and the class of plants for which disturbance decoupling, with two independent sets of outputs, is possible. For Lemma 4.3 to be true, in general the knowledge of plant parameters are needed and tedious computations have to be performed, but then the result holds for a larger class of plants. If Lemma 4.5 is used to satisfy the solvability condition, then only relative degrees of $W_1(s)$, $W_2(s)$ and $W_4(s)$ are needed, but (4.4) is true for a restricted class.

V. CONCLUDING REMARKS

If the initial description of the plant is given in the frequency domain, the solvability condition (4.4) is very simple to check and is also robust with respect to the plant parameters. But the result is not generically true. There is an intuitive interpretation of condition (4.4) in the discrete time case (replace ‘s’ by ‘z’, the forward shift operator, in all transfer matrices). It says that if $E(\cdot)$ and $F(\cdot)$ are nonsingular, the delay in estimating the effect of the disturbance through the measured output, plus the delay in exerting control through the control input, must be at most equal to the delay when the disturbance makes the first point-wise contact with the controlled output in each loop. A causal controller cannot alter this delay relationship.

VI. REFERENCES

[1] Schumacher, J. M., "Compensator Synthesis Based on (C', A, B) - pair," *IEEE Transactions on Automatic Control,* Vol. AC-25, No. 6, pp. 1133-1138, December 1980.

[2] Bhattacharyya, S. P., "Transfer Function Conditions for Output Feedback Disturbance Rejection," *IEEE Transactions on Automatic Control,* Vol. AC-27, No. 4, pp. 974-977, August 1982.

[3] Wolovich, W. A., *Linear Multivariable Systems,* Springer Verlag: New York, 1974.

[4] Morse, A. S., "System Invariants under Feedback and Cascade Control," *Mathematical Systems Theory, Vol. 131,* pp. 61-74, Springer Verlag: New York, 1975.

[5] Hung, N.-T. and B. D. O. Anderson, "Triangularization Technique for the Design of Multivariable Control Systems," *IEEE Transactions on Automatic Control,* Vol. AC-24, No. 3, pp. 455-460, June 1979.

[6] Singh, R. P., *Stable Multivariable Adaptive Control Systems,* Ph.D. Thesis, Yale University, May 1985.

[7] Chen, C.-T., *Introduction to Linear System Theory,* Holt, Rinehart and Winston, Inc.: New York, 1970.

[8] Singh, R. P. and K. S. Narendra, "Prior Information in the Design of Multivariable Adaptive Controllers," *IEEE Transactions on Automatic Control*, Vol. AC-29, No. 12, pp. 1108-1111, December 1984.

[9] Morse, A. S. and W. M. Wonham, "Status of Noninteracting Control," *IEEE Transactions on Automatic Control*, Vol. AC-16, No. 6, pp. 568-581, December 1971.

[10] Warren, M. E. and S. K. Mitter, "Generic Solvability of Morgan's Problem," *IEEE Transactions on Automatic Control*, Vol. AC-20, No. 2, pp. 268-269, April 1975.

[11] Gantmacher, F. R., *The Theory of Matrices, Vol. I*, Chelsea: New York, 1977.

ZERO SPACES, POLE SPACES, DUALS, AND INVERSE SYSTEMS

BOSTWICK F. WYMAN
Department of Mathematics
The Ohio State University
Columbus, Ohio 43210

MICHAEL K. SAIN*
Electrical and Computer Engineering
University of Notre Dame
Notre Dame, Indiana 46556

ABSTRACT

The transfer function $G(s)$ of a finite dimensional, linear, time-invariant dynamical system generates a state space $X(G)$ of poles and a corresponding space $Z(G)$ of zeros. If $G(s)$ has an inverse $\hat{G}(s)$, then $X(\hat{G})$ is comprised algebraically of $Z(G)$ and a variable pole space whose characterization depends upon whether $\hat{G}(s)$ is a left, or a right, inverse. In this note, we unify the left, and right, results by means of appropriate concepts of dual zero spaces, dual pole spaces, and dual variable pole spaces. The method uses module theory for rings which include polynomials, proper transfer functions, and proper stable transfer functions as special cases.

INTRODUCTION

Consider a transfer function $G(s) : U(s) \to Y(s)$ which can be viewed as a pxm matrix of rational functions, or as a linear map between two rational vector spaces. If $G(s)$ is epic (i.e. $G(s)$ has rank p) then $G(s)$ admits right inverses $\hat{G}(s) : Y(s) \to U(s)$. The inverse system is unique if $m = p$; otherwise infinitely many inverses are available. The fundamental pole-zero intuition is that multivariable zeros of $G(s)$ must appear as poles of every inverse system $\hat{G}(s)$, but that under certain circumstances $\hat{G}(s)$ may have additional poles. These results were developed in a very precise algebraic way in [1, 2]. Namely $G(s)$ has a pole space (the minimal realization state space) $X(G)$ and a zero space $Z(G)$ which are both finite dimensional vector spaces with a dynamical action. (See below for definitions or consult the references cited above). Any right inverse $\hat{G}(s)$ has a pole space $X(\hat{G})$ containing a subspace $K(\hat{G})$ of variable poles with corresponding factor space $Z(G) \cong X(\hat{G})/K(\hat{G})$. In fancier language, we have a short exact sequence

$$0 \to K(\hat{G}) \to X(\hat{G}) \to Z(G) \to 0 .$$

Similarly a monic transfer function $H(s)$ with left inverse $\hat{H}(s)$ gives an

---

*The work of the second author was supported by the National Science Foundation under Grant ECS-84-05714.

exact sequence

$$0 \to Z(H) \to X(\hat{H}) \to C(\hat{H}) \to 0 .$$

In [2] these sequences were studied in detail and design limitations on the kernel space $K(\hat{G})$ and the cokernel space $C(\hat{H})$ were presented. The goal of this communication is to describe a comprehensive duality theory which shows that the two sequences above are dual and in some sense cuts the work of certain parts of [2] in half. The duality theory needed is, however, somewhat involved, since we must use duals of finite dimensional vector spaces over the field of scalars, finite dimensional vector spaces over the field of rational functions, and finitely generated free modules over polynomial rings or more complicated rings of rational functions.

TECHNICAL DEVELOPMENT

We will be considering a transfer function, or k(s)-linear mapping

$$G(s) : U(s) \to Y(s) .$$

Here k is a field of scalars and k(s) is the corresponding field of rational functions. To study zeros and poles of G(s) we fix a ring $0 \subset k(s)$. For example, 0 could be the polynomial ring k[s] for the classical theory, the ring 0_∞ of proper transfer functions for the theory "at infinity," or the ring 0_{ps} of proper stable transfer functions. Abstract requirements on 0 are (a) 0 contains the field of scalars; (b) 0 has field of fractions k(s); and (c) 0 is a principal ideal domain. For more about this context, see [2, Sec. VII; 3]. Given 0, and a vector space V over k, we can define the free 0-module

$$\Omega_0 V = 0 \otimes {}_k V .$$

To the map $G(s) : U(s) \to Y(s)$ we can attach an 0-pole module

$$X_0(G) = \Omega_0 U / G^{-1}(\Omega_0 Y) \cap \Omega_0 U ,$$

and an 0-zero module

$$Z_0(G) = \frac{G^{-1}(\Omega_0 Y) + \Omega_0 U}{\ker G + \Omega_0 U} .$$

Next, assume $G(s) : U(s) \to Y(s)$ is epic and $\hat{G}(s) : Y(s) \to U(s)$ is any right inverse. According to [2], there is an exact sequence (labeled E for epic)

$$0 \to K_0(\hat{G}) \to X_0(\hat{G}) \to Z_0(G) \to 0 \qquad \text{(E)}$$

which identifies the zeros of G as essential poles of $\hat{G}$ and defines the variable pole module $K_0(\hat{G})$ as a submodule of $X_0(\hat{G})$.

If $H(s) : U(s) \to Y(s)$ is monic and $\hat{H}(s) : Y(s) \to U(s)$ is any left inverse, there is a corresponding exact sequence (labeled M for monic)

$$0 \to Z_0(H) \to X_0(\hat{H}) \to C_0(\hat{H}) \to 0 \qquad \text{(M)}$$

exhibiting the variable poles $C_0(\hat{H})$ as a factor module.

A main result of [2] was the discovery of restrictions which the variable pole modules must satisfy for every right or left inverse. In fact, these restrictions affect only the Jordan block structure of the variable pole modules and not the numerical locations of the poles themselves. If M is any finitely generated torsion O-module, let t(M), the "torsion rank of M", be the number of invariant factors of M. The number t(M) should be thought of as the smallest number of inputs required to control M or the smallest number of outputs required to observe M (cf., [2, pp. 978-979]). If U has dimension m and Y has dimension p, the results are

$$t\big(K_O(\hat{G})\big) \leq \min(p,\ m-p)$$

$$t\big(C_O(\hat{H})\big) \leq \min(m,\ p-m)\ .$$

The present goal is to show that (E) and (M) are formally dual exact sequences with respect to an appropriately defined duality theory, and that each torsion-rank result above follows from the other by duality. Duality results for pole modules are not surprising. If coordinates are chosen $G(s) = C(sI-A)^{-1}B + D(s)$ becomes a matrix with G^* as transpose. Transposing G(s) also transposes the dynamics matrix A. Duality for zero modules lies deeper since the definition itself is rather involved. In any case, a sophisticated duality theory is required for the present application. The methods used here are quite standard in commutative algebra (see, for example, [4, p. 89]), and many details will be omitted.

Suppose V is a finite dimensional vector space over k and $V(s) = V \otimes {}_k k(s)$ is the extended vector space over k(s). Define

$$V(s)^* = \mathrm{Hom}_{k(s)}\big(V(s),\ k(s)\big)\ ,$$

the space of all k(s) - linear functionals on V(s). If $G(s) : U(s) \to Y(s)$ is a transfer function, then, as usual, $G^* : Y(s)^* \to U(s)^*$ is given by $G^*(\phi) = \phi \circ G$ for $\phi : Y(s) \to k(s)$ in $Y(s)^*$. If appropriate bases and dual bases are chosen, the matrix of G^* is the transpose of the matrix of G. Furthermore, if $V^* = \mathrm{Hom}_k(V,\ k)$ is the k-dual of V, then $V^*(s) = V^* \otimes {}_k k(s)$ can be identified with $V(s)^*$.

Next, suppose $L \subset V(s)$ is a free O-submodule of full rank. Then the annihilator of L in $V(s)^*$ is defined to be $L^* = \{\phi \in V(s)^* : \phi(v) \in O$ for all v in $L\}$. Typically $L = \Omega_O V \subset V(s)$, and we write $\Omega_O{}^*V = (\Omega_O V)^*$. In fact, we can identify $\Omega_O{}^*V = \Omega_O V^* \subset V^*(s)$.

In addition to duals of k(s) - vector spaces, we also need to consider duals of finitely generated torsion O-modules.

If X is such a module, define

$$X^* = \mathrm{Hom}_O\Big(\ X,\ \frac{k(s)}{O}\ \Big)\ .$$

Since X is a finite-dimensional vector space over k, the space $\mathrm{Hom}_k(X,\ k)$ is another reasonable dual, at least on the vector space level. In fact, these two approaches coincide, since

$$\mathrm{Hom}_k(X,\ k) \cong \mathrm{Hom}_O\Big(\ X,\ \frac{k(s)}{O}\ \Big)$$

as vector spaces over k. If O = k[s], this isomorphism is implicit in abstract realization theory [5], namely, a scalar output $C : X \to k$ extends to a module output $\tilde{C} : X \to k(s)/k[s]$. For more general rings it is best to localize and apply [6, Ch.4]. As a corollary we conclude that duality is reflexive $\{X^{**} \cong X\}$ and exact (the dual of an exact sequence is exact).

Finally we are ready to state the main results on the dualization of poles and zeros.

Proposition 1. (Duality of Pole Spaces) Suppose $G(s) : U(s) \to Y(s)$ is a transfer function with dual $G^*(s) : Y(s)^* \to U(s)^*$. Let $X_O(G)$ and $X_O(G^*)$ be the corresponding pole modules over the ring O. Then there is a nonsingular O-module pairing

$$X_O(G) \times X_O(G^*) \to \frac{k(s)}{O}$$

which establishes an O-module isomorphism

$$X_O(G)^* \cong X_O(G^*) .$$

Idea of proof. The pairing (we suppress the subscript O)

$$\frac{\Omega U}{G^{-1}(\Omega Y) \cap \Omega U} \times \frac{\Omega^* Y}{G^{*-1}(\Omega^* U) \cap \Omega^* Y} \to \frac{k(s)}{O}$$

is defined to be the map induced from

$$\Omega U \times \Omega^* Y \to k(s)$$

$$(u, \phi) = \phi(Gu) = (G^*\phi)u .$$

We see easily that $(u, \phi) = 0$ for all u in $G^{-1}(\Omega Y) \cap \Omega U$ and all ϕ in $\Omega^* Y$. Similarly $(u, \phi) = 0$ for all u in ΩU and ϕ in $G^{*-1}(\Omega^* U) \cap \Omega^* Y$.

Proposition 2. (Duality of Zero Spaces) If $G(s)$ and $G^*(s)$ are as in Proposition 1, then there is a nonsingular O-module pairing

$$Z_O(G) \times Z_O(G^*) \to \frac{k(s)}{O}$$

which establishes an O-module isomorphism $Z_O(G^*) \cong Z_O(G)^*$.

Idea of proof. The required pairing (again we suppress the subscript O) is induced from the k(s) - pairing

$$\left(G^{-1}(\Omega Y) + \Omega U\right) \times \left(G^{*-1}(\Omega^* U) + \Omega^* Y\right) \to k(s)$$

defined by

$$(u, \phi) = \phi\left(G(s)u(s)\right) = \left(G^*\phi\right)\left(u(s)\right)$$

as before. This time we must make the technical verification that $G^*\phi\left(u(s)\right)$ lies in O if and only if $u(s)$ is in $\ker G(s) + \Omega U$.

Finally, assume $G(s) : U(s) \to Y(s)$ is epic, with $\dim U = m$ and $\dim Y = p$ over k. Let $\hat{G}$ be any right inverse and consider the exact sequence of O-modules

$$0 \to K_O(\hat{G}) \to X_O(\hat{G}) \to Z_O(G) \to 0 \; . \qquad (E)$$

Dualizing, we obtain the exact sequence

$$0 \to Z_O(G)^* \to X_O(\hat{G})^* \to K_O(\hat{G})^* \to 0 \; . \qquad (E^*)$$

On the other hand, since $(G\hat{G})^* = \hat{G}^*G^*$, it follows that G^* is monic and $\hat{G}^*$ is one of its left inverses. That is, we have an exact sequence for the monic case:

$$0 \to Z_O(G^*) \to X_O(\hat{G}^*) \to C_O(\hat{G}^*) \to 0 \; . \qquad (\hat{M})$$

Proposition 3. (Duality of Variable Pole Modules) The exact sequences (E^*) and $(\hat{M})$ above are isomorphic exact sequences. In particular, $K_O(\hat{G})^*$ and $C_O(\hat{G}^*)$ are isomorphic.

Idea of Proof. We already know that $X_O(\hat{G}^*)$ and $X_O(\hat{G})^*$ are isomorphic (Proposition 1) and $Z_O(G)^*$ and $Z_O(G^*)$ are isomorphic (Proposition 2). Checking that these isomorphisms are consistent insures the existence of an induced isomorphism between $K_O(\hat{G})^*$ and $C_O(\hat{G}^*)$.

Now, suppose that the torsion rank result referred to above has been established for epic $G(s) : U(s) \to Y(s)$. Namely, assume proved that $t\left(K_O(\hat{G})\right) \leq \min(p, m-p)$ for every (monic) right inverse $\hat{G}$ of G. We claim that the corresponding result for monic maps follows from the duality in Proposition 3. Let $H(s) : U(s) \to Y(s)$ be monic, so that $H^*(s) : Y^*(s) \to U^*(s)$ is epic. Note that the role of m and p is also interchanged. Let $\hat{H}$ be any left inverse of H, so that by the epic result, $t\left(K_O(\hat{H}^*)\right) \leq \min(m, p-m)$. Apply Proposition 3 with $H^* = G$ and identify $\hat{H}^{**} = \hat{H}$, so that $K_O(\hat{H}^*)$ is dual to $C_O(\hat{H})$. Since dual finitely generated torsion modules are isomorphic by structure theory over principal ideal domains, it follows that $t\left(C_O(\hat{H})\right) = t\left(K_O(\hat{H}^*)\right)$. Thus $t\left(C_O(\hat{H})\right) \leq \min(m, p-m)$ as required.

REFERENCES

1. B.F. Wyman and M.K. Sain, "The Zero Module and Essential Inverse Systems", IEEE Transactions on Circuits and Systems, Vol. CAS-28, No. 2, pp. 112-126, February 1981.
2. B.F. Wyman and M.K. Sain, "On the Design of Pole Modules for Inverse Systems", IEEE Transactions on Circuits and Systems, Vol. CAS-32, No. 10, pp. 977-988, October 1985.
3. B.F. Wyman, G. Conte, and A.M. Perdon, "Local and Global Linear System Theory", Proceedings Int. Symp. Mathematical Theory Networks and Systems, Stockholm, 1985; to appear.
4. C.W. Curtis and I. Reiner, Methods of Representation Theory, Vol. I. Wiley: New York, 1981.
5. B.F. Wyman, "Linear Systems over Rings of Operators", in: E. Manes, Ed., Category Theory Applied to Computation and Control, Lec. Notes in Comp. Sci., Vol. 25. Springer-Verlag: Berlin, 1975, pp. 218-223.
6. R. Hartshorne, Local Cohomology, Lecture Notes in Mathematics, Vol. 41. Springer-Verlag: Berlin, 1967.

SQUARING DOWN BY STATIC AND DYNAMIC COMPENSATORS

A. SABERI
Department of Electrical and
Computer Engineering
Washington State University
Pullman, WA 99164-2752

P. SANNUTI
Department of Electrical Engineering
P. O. Box 909
Rutgers University
Piscataway, NJ 08854

ABSTRACT

Given a system, which is not necessarily invertible and which has unequal number of inputs and outputs, a method of "squaring down," that is a method of designing pre and post compensators such that the resulting system has equal number of inputs and outputs and is invertible, is presented. The compensators designed are in general dynamic and have the property that the additional finite zeros induced by them are assignable to the open left half complex plane. Moreover, our compensator design procedure preserves the stabilizability, detectability and minimum phase properties, and the infinite zero structure of the original system. Thus, the paper opens up a new method of designing nonsquare systems by converting them to square invertible systems. Most of the frequency domain or time domain design schemes that has hither to been limited to only square invertible systems can now be extended to nonsquare and not necessarily invertible systems.

1. INTRODUCTION

In the design of multivariable feedback systems, often feedback loops are introduced between a selected set of measured output variables and an equal number of independent control inputs. Thus, given a system with unequal number of inputs and outputs (nonsquare system), the first stage of design essentially consists of combining all outputs together into a new set of outputs, and/or similarly combining all inputs together into a new set of inputs such that the resulting system has equal number of inputs and outputs and is invertible. Also, since invertibility is used as a starting assumption in a majority of design schemes, even if the given system is square but not invertible, one needs an initial design step of reducing both the number of inputs and outputs in order to make the resulting system invertible. Such as initial design problem can be called a "squaring down" problem. Squaring down a given system with transfer matrix H(s), is essentially equivalent to finding post or/and pre compensators, K_{post} and K_{pre}, such that $K_{post}\ H(s)\ K_{pre}$ is a square invertible system.

Several aspects of the squaring down problem have already been studied by several researchers [1,2,4-6,11,12]. It is known that the solution of the squaring down problem has significant consequences on the zero structure of the resulting system and hence vitally affects the final design procedure. In fact, the squared down system always has additional set of finite zeros apart from the zeros (if any) inherited from the original nonsquare system. It is imperative that these additional zeros be placed at the desired locations in the open left half complex plane by an appropriate choice of pre and post compensators.

Recently, Sannuti and Saberi [10] showed that any given linear system can be represented by a special coordinate basis (s.c.b.) which explicitly pinpoints the finite and infinite zero structure of a given system. In the light of such a s.c.b., we first isolate a subsystem (or subsystems) whose pole assignment by constant output feedback is equivalent to the assignment of additional zeros in the squared down system. Next, we investigate the use of dynamic compensators for solving the squaring down problem. We construct minimal order dynamic compensators, K_{post} and K_{pre}, such that the added finite zeros are always assigned to the open left half complex plane. Any stabilizable, detectable, non square and not necessarily invertible system can be squared down via dynamic compensators. Except for an obvious reduction of the number of inputs or/and outputs, our compensator design doesn't destroy any fundamental features of a given system. That is if the given system is stabilizable, detectable and of minimum phase, so is the squared invertible system. Also, the infinite zero structure of the squared down system is the same as that of the original system. Thus, this paper provides a key in extending several existing design schemes which are normally meant for square invertible systems to nonsquare and not necessarily invertible systems. We emphasize this aspect of our work by mentioning three applications: (1) Diagonal decoupling with state feedback, (2) Almost disturbance decoupling with output feedback and (3) decentralized stabilization with output feedback. Throughout this paper, prime indicates a transpose, I_k denotes the identity matrix of dimension k, $\lambda(A)$ denotes the eigenvalues of A and C^- denotes the open left half complex plane. Due to lack of space, we present here our essential results. Proofs of various theorems and other clarifications of our results appear elsewhere.

2. FINITE AND INFINITE ZERO STRUCTURE--A SPECIAL COORDINATE BASIS

Consider a linear system $\hat{\Sigma}$,

$$\dot{\hat{x}} = \hat{A}\,\hat{x} + \hat{B}\,\hat{u} \quad , \quad \hat{y} = \hat{C}\,\hat{x} \tag{2.1}$$

where $\hat{x}$, $\hat{y}$ and $\hat{u}$ are respectively n-, p- and m-dimensional vectors. Assume that $\hat{B}$ and $\hat{C}$ are of full rank. It is shown in [10] that (2.1) can be transformed to a special coordinate basis (s.c.b.) which explicitly displays the finite and infinite zero structure of it. Theorem 2.1 gives the structure of s.c.b. (see also Figure 2.1)

Theorem 2.1: There exist nonsingular transformations Γ_1, Γ_2, and Γ_3, an integer $K \le n$, and integer indices q_i and r_i, $i = 1$ to K, such that

$$\hat{x} = \Gamma_1 x,\ \hat{y} = \Gamma_2[y_f', y_s']' \ , \quad \hat{u} = \Gamma_3[u' \ , v']'$$

$$u = [u_1' \ , u_2', \ldots u_K']' \quad , \quad x = [x_a', x_b', x_c', x_f']'$$

$$x_{if} = [x_{1i-1}', x_{2i-2}', \ldots x_{i0}']' \quad , \quad x_f = [x_{1f}', x_{2f}', \ldots x_{Kf}']'$$

$$x_{bis} = [x_{b1i-1}', x_{b2i-2}', \ldots x_{bi0}']' \ , x_b = [x_{b1s}' \ , x_{b2s}', \ldots x_{bKs}']'$$

$$\dot{x}_a = A_{aa}x_a + A_{af}y_f + A_{as}y_s \tag{2.2a}$$

$$\dot{x}_b = A_{bb}x_b + A_{bf}y_f \tag{2.2b}$$

$$\dot{x}_c = A_{cc}x_c + B_c v + A_{ca}x_a + A_{cs}y_s + A_{cf}y_f \tag{2.2c}$$

$$\dot{x}_{if} = A_{if}x_{if} + L_{if}y_f$$

$$+ B_{if}[u_i + E_{ia}x_a + E_{ib}x_b + E_{ic}x_c + \sum_{j=1}^{K} E_{ij}x_{jf}] \tag{2.2d}$$

$$y_{if} = C_{if}x_{if} = x_{1i-1},\ y_f = C_f x_f, \tag{2.2e}$$

$$y_{is} = C_{is}x_{bis} = x_{b1i-1},\ y_s = C_s x_b. \tag{2.2f}$$

Here, the last q_i rows of L_{if} have zero elements and that $A_{1f} = 0$,

$$A_{if} = \begin{bmatrix} 0 & I_{(i-1)q_i} \\ \\ 0 & 0 \end{bmatrix}, \quad B_{if} = \begin{bmatrix} 0 \\ \\ I_{q_i} \end{bmatrix},$$

$$C_f = [C_{1f}, C_{2f}, \ldots C_{Kf}] \ , \ C_{if} = [I_{q_i} \ , \ 0]$$

$$C_s = [C_{1s}, C_{2s}, \ldots C_{Ks}] \ , \ C_{is} = [I_{r_i} \ , \ 0] \ , \quad i = 1 \text{ to } K.$$

Furthermore, (A_{cc}, B_c) is a controllable pair.

For clarity, the dimension of each variable is given now: x_a, x_b, x_{bis}, x_{bij}, x_c, x_f, x_{if}, x_{ij}, u, u_i, v, y_f, y_{if}, y_s and y_{is} are of dimension n_a, n_b, ir_i, r_{i+j}, n_c, n_f, n_{if}, q_{i+j}, m_u, q_i, m_v, p_f, q_i, p_s and r_i respectively, where

$$p = \Sigma_{i=1}^{K} (q_i + r_i), \ m_u = p_f = \Sigma_{i=1}^{K} q_i, \ m_v = m - m_u \ , \ p_s = \Sigma_{i=1}^{K} r_i = p - p_f,$$

$$n_{if} = iq_i, \ n_f = \Sigma_{i=1}^{K} n_{if}, \ n_b = \Sigma_{i=1}^{K} ir_i, \ n = n_a + n_b + n_c + n_f.$$

Several important properties of s.c.b. which reveal the zero structure of the given system presented below.

Property 2.1: $\hat{\Sigma}$ is observable (detectable) iff (A_{obs}, C_{obs}) is a observable (detectable) pair, where

$$A_{obs} = \begin{bmatrix} A_{aa} & 0 \\ \\ A_{ca} & A_{cc} \end{bmatrix}, \quad C_{obs} = [E_a, E_c]$$

$$E_a = [E'_{1a}, E'_{2a}, \ldots E'_{Ka}]' \ , \ E_c = [E'_{1c}, E'_{2c}, \ldots E'_{Kc}]'.$$

$\hat{\Sigma}$ is controllable (stabilizable) iff (A_{con}, B_{con}) is a controllable (stabilizable) pair, where

$$A_{con} = \begin{bmatrix} A_{aa} & A_{as}C_s \\ \\ 0 & A_{bb} \end{bmatrix}, \quad B_{con} = \begin{bmatrix} A_{af} \\ \\ A_{bf} \end{bmatrix}$$

Property 2.2: System $\hat{\Sigma}$ is right invertible iff x_b and hence y_s is nonexistent ($n_b = 0$, $p_s = 0$), left invertible iff x_c and hence v is nonexistent ($n_c = 0$, $m_v = 0$), invertible iff both x_b and x_c are nonexistent. Furthermore, $\hat{\Sigma}$ is degenerate iff both n_b and n_c are non zero (i.e., both states x_b and x_c are present in the s.c.b.).

Property 2.3: Invariant zeros [6] of $\hat{\Sigma}$ are the eignevalues of A_{aa}.

Under the assumption that $\hat{\Sigma}$ is nondegenerate and it is controllable and observable, invariant zeros are same as the transmission zeros of Davison and Wang [3]. In view of this, we say that $\hat{\Sigma}$ is of minimum Phase if $\lambda(A_{aa})$ are in C^-. Let us adopt the definition of orders of infinite zeros as defined in Smith-McMillan theory [8].

Property 2.4: There are iq_i number of infinite zeros of order i, for i = 1 to K. Here q_i is as in our s.c.b. and is the dimension of u_i as well as that of y_{if}. We note that $\hat{\Sigma}$ does not have an infinite zero of order j iff $q_j = 0$.

3. SQUARING DOWN

3.1 Left Invertible Systems

For left invertible systems, as property 2.2 implies, the state x_c and hence control v is nonexistent and the given system is nondegenerate. Furthermore, the dimension of u is same as that of y_f. In order to square down the output variables, we need to design a post compensator, K_{post}. For a static compensator, we need to generate a new output vector $\tilde{y}_f$ by linearly combining y_f and y_s,

$$\tilde{y}_f = K_{post}[y_f', y_s']' = L_a y_f + L y_s .$$

When L_a is a singular matrix, it can be easily shown that the structure at infinity of the resulting system is different from that of the given system. Hence, in order to preserve the structure at infinity, we take L_a as a nonsingular matrix. Without loss of generality, L_a then can be taken as an identity matrix. The matrix L in general cannot be taken as a zero matrix, since then it amounts to completely neglecting y_s as an output variable. Furthermore, in the case when L = 0, the status of state x_b is same as that of x_a, implying that $\lambda(A_{bb})$ are additional invariant zeros. In general, $\lambda(A_{bb})$ need not be in C^- and hence our design problem is to choose L such that the additional invariant zeros induced due to squaring down are in C^-. Choosing

$$\tilde{y}_f = K_{post}[y_f', y_s']' , \quad K_{post} = [I_{p_f} , L], \tag{3.1}$$

and letting $\tilde{x}_a = [x_a', x_b']'$, (2.2a, b) can be rewritten as

$$\dot{\tilde{x}}_a = \tilde{A}_{aa}\tilde{x}_a + \tilde{A}_{af}\tilde{y}_f , \tag{3.2}$$

$$\tilde{A}_{aa} = \begin{bmatrix} A_{aa} & A_{as}C_s - A_{af}LC_s \\ 0 & A_{bb} - A_{bf}LC_s \end{bmatrix} , \quad \tilde{A}_{af} = \begin{bmatrix} A_{af} \\ A_{bf} \end{bmatrix} . \tag{3.3}$$

Equations (3.2) and (2.3d) with $x_c = 0$ can be considered as the new dynamic equations. It can be easily shown [10] that the new invariant zeros are given by $\lambda(\tilde{A}_{aa})$ which are $\lambda(A_{aa})$ and $\lambda(A_{bb}-A_{bf}LC_s)$. This discussion leads to theorem 3.1.

Theorem 3.1: The post compensator (3.1) induces an additional set of invariant zeros in the squared down system where $\tilde{y}_f$ is the new output. These additional invariant zeros are given by $\lambda(A_{bb}-A_{bf}LC_s)$.

Since additional invariant zeros are $\lambda(A_{bb}-A_{bf}LC_s)$, the problem of additional invariant zero assignment by an appropriate choice of L, is the same as the problem of pole assignment by output feedback of a system whose matrix triple is (A_{bb}, A_{bf}, C_s). It is not always possible to select L in order to assign the additional invariant zeros to C^-. This implies that squaring down with the constraint of additional invariant zeros being in C^- cannot in general be accomplished with static compensators alone. We are thus lead to the use of dynamic compensators.

In designing the required dynamic compensators we need certain results from observer theory. We recall the following lemma for linear state function observers which is an excerpt from [7].

Lemma 3.1. Consider a stabilizable and detectable system

$$\dot{x} = Ax + Bu \ , \quad y = Cx,$$

where x, u, y are of dimensions n, m, and p respectively. Also let $u = -Fx$ be a desired state feedback control law and $\lambda(A-BF)$ belongs to C^-. A minimal order observer is characterized by matrices W, N, and M of dimensions $(n-p)xn$, $(n-p) \times (n-p)$ and $(n-p)xp$, such that

(i) $WA = NW + MC$

(ii) $\text{rank}\ [W'C'] = n$

(iii) $\lambda(N) \subset C^-$.

Then, the observer-compensator realizing the state feedback control law $u = -Fx$ is given by

$$\dot{z} = (N-WBG)z + (M-WBJ)y$$

$$u = -Gz - Jy,$$

where [G,J] is given by

$$[G,J]' = [W'\ C']^{-1}\ F'.$$

We now proceed to the design of dynamic post compensators. Throughout this paper we will assume that $\hat{\Sigma}$ is stabilizable and detectable. Then by property 2.1, the pair (A_{bb}, A_{bf}) is stabilizable and the pair (A_{bb}, C_s) is observable. As we commented earlier, "squaring down" adds additional invariant zeros to the resulting system. Let C_d^- be the desired locations for these additional invariant zeros. Also let F and N be such that $\lambda(A_{bb}-A_{bf}F)$ and $\lambda(N)$ are in C_d^-. We note that, since (A_{bb}, A_{bf}) is in general only stabilizable, those $\lambda(A_{bb})$ which are stable but not controllable must be included in C_d^-. Now employing lemma 3.1 and using (A_{bb}, A_{bf}, C_s) in place of (A,B,C), calculate the matrices W, M, G, and J. Finally, define a dynamic post compensator K_{post}

$$\dot{z} = Nz + My_s + WA_{bf}y_f \quad , \quad \tilde{y}_f = y_f + Jy_s + Gz \tag{3.4}$$

Here z is an (n_b-p_s) dimensional state which represents the dynamics of K_{post}. Let the transfer function of K_{post} be denoted by $H_{post}(s)$. We have the following theorem 3.2.

Theorem 3.2: Consider a left invertible, stabilizable and detectable system $\hat{\Sigma}$ with transfer function $\hat{H}(s)$. Define a square system Σ_{sq} with transfer function $H_{sq}(s)$ as

$$H_{sq}(s) = H_{post}(s)\Gamma_2^{-1}\hat{H}(s)\Gamma_3,$$

where Γ_2 and Γ_3 are as in theorem 2.1 and K_{post} is as defined by (3.4). Note that Σ_{sq} is formed by cascading Γ_3, $\hat{\Sigma}$, Γ_2^{-1} and K_{post} in the order given. Then Σ_{sq} has the following properties:

(i) Σ_{sq} is invertible.

(ii) Invariant zeros of Σ_{sq} = Invariant zeros of $\hat{\Sigma}$(i.e., $\lambda(A_{aa})$) + $\lambda(N)$ + $\lambda(A_{bb}-A_{bf}F)$.

(iii) The dynamic order of Σ_{sq} is $n + n_b - p_s$.

(iv) The structure at infinity of Σ_{sq} is same as that of $\hat{\Sigma}$.

Remark 3.1: If an L exists such that $\lambda(A_{bb}-A_{bf}LC_s)$ are at the required locations in C^-, then $F = LC_s$, $J = L$ and $G = 0$. Thus, in this case, the dynamic compensator K_{post} reduces to a static compensator.

Remark 3.2: $\lambda(N)$ are among the eignevalues of Σ_{sq}. These modes are uncontrollable in Σ_{sq}, however, they may be observable in Σ_{sq}. The fact that $\lambda(N)$ are uncontrollable in Σ_{sq} is not detrimental to overall system design because $\lambda(N)$ have already been placed at the desired locations in C^- while designing K_{post}.

Remark 3.3: If $\hat{\Sigma}$ is stabilizable, detectable, and of minimum phase, so is Σ_{sq}. This can be easily seen as follows: $\hat{\Sigma}$ is of minimum phase implies that $\lambda(A_{aa})$ are in C^-. Thus, all the invariant zeros of Σ_{sq} are in C^- implying that Σ_{sq} is of minimum phase. The fact that Σ_{sq} is of minimum phase and invertible leads to the conclusion that it is stabilizable and detectable [2].

3.2. Right Invertible Systems

We now consider a right invertible $\hat{\Sigma}$. In this case, x_b and hence y_s

is nonexistent and $\hat{\Sigma}$ is nondegenerate. Furthermore, the dimension of u is same as that of y_f. In order to square down the input variables, we need to design a pre compensator K_{pre}, such that controls u and v can be generated from a new control $\tilde{u}$. Following an analogous treatment as in the left invertible case, let

$$[u',v']' = K_{pre}\tilde{u} \quad , \quad K_{pre} = [I_m \; ,\tilde{L}']', \qquad (3.5)$$

we have the following theorem 3.3.

Theorem 3.3: The pre compensator, K_{pre} of (3.7) induces an additional set of invariant zeros in the squared down system where $\tilde{u}$ is the new input variable. These additional invariant zeros are given by $\lambda(A_{cc} - B_c\tilde{L}E_c)$.

The problem of additional invariant zero assignment by an appropriate choice of $\tilde{L}$, is same as the pole assignment by output feedback of a system whose matrix triple is (A_{cc}, B_c, E_c). Thus, as in the left invertible case, the additional invariant zeros cannot always be assignable to C^- by a static pre compensator. The remedy, of course, then as in the previous case, is to use a dynamic compensator.

Let the rank of $m_u \times n_c$ matrix E_c be p_c where obviously $p_c \leq \min(m_u, n_c)$. Let Φ be a $m_u \times m_u$ non-singular row compression matrix such that

$$\Phi E_c = [\bar{E}_c', 0]' \quad , \text{Rank } \bar{E}_c = p_c .$$

Let $\tilde{F}$ and $\tilde{N}$ be such that $\lambda(A_{cc}-B_c\tilde{F})$ and $\lambda(\tilde{N})$ are in C_d^-. Now employing lemma 3.1 and using $(A_{cc}, B_c, \bar{E}_c)$ in place of (A, B, C), calculate the matrices $\tilde{W}$, $\tilde{M}$, $\tilde{G}$, and $\tilde{J}$. (The notational change is obvious, $\tilde{F}$, $\tilde{N}$, ... replace F, N, ...). Noting that $\tilde{M}$ and $\tilde{J}$ thus calculated are of dimension $(n_c - p_c) \times p_c$ and $m_v \times p_c$ respectively, we define matrices $\bar{M}$ and $\bar{J}$ of dimension $(n_c - p_c) \times m_u$ and $m_v \times m_u$ by augmenting appropriate number of zero column to $\tilde{M}$ and $\tilde{J}$,

$$\bar{M} = [\tilde{M}, 0] \quad , \quad \bar{J} = [\tilde{J}, 0].$$

Next define a dynamic pre compensator K_{pre} as

$$\dot{\tilde{z}} = (\tilde{N} - \tilde{W}B_c\tilde{G})\tilde{z} - (\bar{M} - \tilde{W}B_c\bar{J})\Phi\tilde{u} \quad , v = -\tilde{G}\tilde{z} + \bar{J}\Phi\tilde{u} \quad , \quad u = \tilde{u} \qquad (3.6)$$

Here $\tilde{z}$ is an (n_c-p_c) dimensional state which represents the dynamics of K_{pre}. Let the transfer function of K_{pre} by denoted by $H_{pre}(s)$. We have the following theorem 3.4.

Theorem 3.4: Consider a right invertible, stabilizable and detectable system $\hat{\Sigma}$ with transfer function $\hat{H}(s)$. Define a square system Σ_{sq} with the transfer function $H_{sq}(s)$ as

$H_{sq}(s) = \Gamma_2^{-1}\hat{H}(s)\Gamma_3 H_{pre}(s)$

where Γ_2 and Γ_3 are as in theorem 2.1, and K_{pre} as defined by (3.6). Note that Σ_{sq} is formed by cascading K_{pre}, Γ_3, $\hat{\Sigma}$ and Γ_2^{-1} in the order given. Then Σ_{sq} has the following properties:

(i) Σ_{sq} is invertible.

(ii) Invariant zeros of Σ_{sq} = Invariant zeros of $\hat{\Sigma}$(i.e., $\lambda(A_{aa})$) + $\lambda(\tilde{N})$ + $\lambda(A_{cc} - B_c\tilde{F})$)

(iii) The dynamic order of Σ_{sq} is $n+n_c-p_c$ where p_c is the rank of E_c.

(iv) The structure at infinity of Σ_{sq} is same as that of $\hat{\Sigma}$.

Remark 3.4: If an $\tilde{L}$ exists such that $\lambda(A_{cc} - B_c\tilde{L}E_c)$ are at the required locations in C_d^-, then $\tilde{F} = \tilde{L}E_c$, $\tilde{J}\ \Phi = \tilde{L}$ and $\tilde{G} = 0$. Thus, in this case, the dynamic compensator K_{pre} reduces to a static compensator.

Remark 3.5: If the given system $\hat{\Sigma}$ is stabilizable, detectable and of minimum phase, so the Σ_{sq}.

Remark 3.6: Although there is a good amount of similarity in the design and implementation of K_{post} and that of K_{pre}, one cannot say that there is a complete dualism between them. We point out that $\lambda(N)$ are among the eigenvalues of Σ_{sq} of section 3.1 and they are not controllable in Σ_{sq}, although they may be observable. However, $\lambda(\tilde{N})$ need not be among the eigenvalues of Σ_{sq} of this section; furthermore, Σ_{sq} of this section could be controllable and observable.

3.3 Neither left nor right invertible systems

We can now consider the general squaring down problem where the given system $\hat{\Sigma}$ is neither left nor right invertible. In this case, both x_b and x_c are present in the s.c.b. of $\hat{\Sigma}$ and hence $\hat{\Sigma}$ is degenerate. This necessitates the use of both pre and post compensators in order to obtain a square invertible system. Our design philosophy is straight forward. We first obtain a s.c.b. of $\hat{\Sigma}$. Using the triple (A_{bb}, A_{bf}, C_s), a post compensator K_{post} is designed next as in section 3.1. Then using the triple (A_{cc}, B_c, E_c), a pre compensator K_{pre} is designed as in section 3.2.

Remark 3.7: In some applications, one does not need complete squaring down. Only, either left or right invertibility is necessary. If the given system is neither left nor right invertible, use of only K_{post} will make the system right invertible and analogously, use of only K_{pre} will make it left invertible.

We want to point out that even if the given $\hat{\Sigma}$ is square, it might not be invertible. Thus, being just square is not the main consideration. The important point is to construct pre and post compensators such that the resulting squared system is invertible. Furthermore, one should preserve the fundamental properties such as stabilizability, detectability, minimum phase nature and infinite zero structure of $\hat{\Sigma}$. This is exactly what our compensator design achieves. Thus, our initial compensator design facilitates the final design of dynamical systems by converting them to square invertible systems.

4. APPLICATIONS

There are a number of design schemes which require "squaring down" as a first step before a final design is accomplished. Diagonal decoupling, almost disturbance decoupling with output feedback and decentralized stabilization with output feedback are three such design schemes. Due to lack of space we will describe only one of these . In [13], for square invertible and minimum phase plants, Saberi developed an explicit design scheme for a feedback controller which utilizes only output feedback and which guarantees internal stability while rejecting or decoupling any disturbance (an element of L_p space) to any given accuracy. In view of this paper the square and invertible assumption of [13] can be removed and any minimum phase plant can be considered.

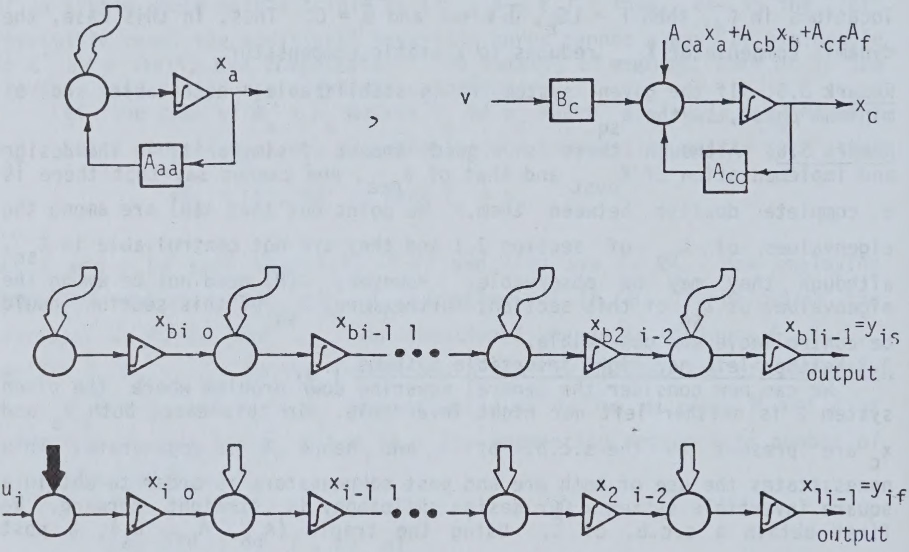


Figure 2-1: A block diagram representation of system $\hat{\Sigma}$. Here a signal given by the solid lined arrow ⬇ is some linear combination of all states, a signal given by a thin lined arrow ⇩ is some linear combination of only outputs y_f, where as a signal given by a thin curved lined arrow ⤵ is some linear combination of all the outputs y_f and y_s.

REFERENCES

[1] Aplevich, J.D., "On the Squaring Problem in Feedback Design," 1981, JACC Charlottesville, WP-4P.

[2] Davison, E.J. "Some Properties of Minimum Phase Systems and Squared-Down Systems," IEEE Trans. Auto. Control, AC-28: p.221-222, 1983.

[3] Davison, E.J. and Wang, S.H. "Properties and Calculations of Transmission Zeros of Linear Multivariable Systems," Automatica, 1974, p.643-658.

[4] Karcanias, N. and Giannaropoulos, C., "Grassman Invariants, Almost Zeros and the Determinantal Zero, Pole Assignment Problems of Linear Multivariable Systems," 1984, Int. J. Control, Vol. 40, p. 673-698.

[5] Kouvaritakis, B. and MacFarlane, A.G.J., "Geometric Approach to Analysis and Synthesis of System Zeros: Part 2, Nonsquare Systems," Int. J. Control, 23:p.167-181, 1976.

[6] MacFarlane, A.G.J. and Karcanias, N. "Poles and Zeros of Linear Multivariable Systems: A Survey of the Algebraic, Geometric and Complex-Variable Theory," Int. J. Control, 24:p.3374, 1976.

[7] O'Reilly, J., Observers for Linear Systems, Academic Press, 1983.

[8] Rosenbrock, H.H., State-Space and Multivariable Theory, John Wiley, New York, 1970.

[9] Rosenbrock, H.H., "The Zeros of a System," Int. J. Control, 18:p.297-299, 1973.

[10] Sannuti, P. and Saberi, A., "A Special Coordinate Basis of Multivariable Linear Systems-Finite and Infinite Zero Structure, Squaring Down and Decoupling," Technical Report, Rutgers University, 1985. To appear in Int. J. Control.

[11] Vardulakis, A.I.G., "On Infinite Zeros," Int. J. Control, 32:p.849-866, 1980.

[12] Vardulakis, A.I.G., "Zero Placement and the 'Squaring Down' Problem: A Polynomial Matrix Approach," Int. J. Control 31:821-832, 1980.

[13] Saberi, A., "Output Feedback Control with Almost Disturbance Decoupling Property: A Singular Perturbation Approach," in Proc. 23rd Conf. Decision Contr., Las Vegas, NV, pp. 625-631, 1984.

ASYMPTOTIC FORMS OF THE GENERALIZED HESSENBERG REPRESENTATION

D. K. LINDNER and D. W. LUSE
Department of Electrical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
(703)961-5114

The following situation occurs often when an experimentally derived mathematical system model is transformed into a specific form, such as the Generalized Hessenberg Representation (GHR) [3], through a computational procedure. It will be recognized that some model simplification may be possible due to the presence of very large and/or small numbers in the specific system form. The natural procedure in such situations is to replace the large/small quantities by variables, and to study the effect of allowing the variables to approach infinity/zero. This brings the problem within the realm of perturbation theory, and methods such as those described in [1] may be used.

It is demonstrated in [1] that system matrix techniques can be useful in carrying out the required limiting processes. These system matrix methods are developed to a greater degree in [2], where modes lost in the limiting process are carefully accounted for. In this paper, we will demonstrate the process for the case when the GHR has a super-diagonal block, all of whose elements are small. Thus, we suppose that a system has been transformed to the following GHR form, and that the block E_{23} is small.

$$\dot{x} = \begin{bmatrix} A_{11} & A_{12} & 0 & 0 \\ A_{21} & A_{22} & E_{23} & 0 \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} x + \begin{bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix} u \qquad (1)$$

$$y = [\, C \quad 0 \quad 0 \quad 0 \,]\, x$$

Following the usual procedure, the block E_{23} is replaced by εA_{23}. We now write the system in system matrix form [4]:

$$\begin{bmatrix} sI-A_{11} & -A_{12} & 0 & 0 & B_1 \\ -A_{21} & sI-A_{22} & -\varepsilon A_{23} & 0 & B_2 \\ -A_{31} & -A_{32} & sI-A_{33} & -A_{34} & B_3 \\ -A_{41} & -A_{42} & -A_{43} & sI-A_{44} & B_4 \\ -C & 0 & 0 & 0 & 0 \end{bmatrix}$$

Two things now become apparent:

A) The system poles approach the eigenvalues of the matrices

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad \begin{bmatrix} A_{33} & A_{34} \\ A_{43} & A_{44} \end{bmatrix}$$

B) The eigenvalues of the second matrix above are becoming asymptotically unobservable as ε becomes small, and an approximate transfer matrix can be computed from the reduced system matrix

$$\begin{bmatrix} sI-A_{11} & -A_{12} & B_1 \\ -A_{21} & sI-A_{22} & B_2 \\ -C & 0 & 0 \end{bmatrix}$$

Of course, this reduced-order system is useful for control design only when the discarded modes are stable.

Although [2] concerns two-frequency-scale transfer matrices, regular perturbations are accounted for also. Thus, in the language of [2], the system (1) is "one frequency scale" and the eigenvalues mentioned in B) above are the "lost (slow) poles." A similar analysis applies to other limiting forms of the GHR, such as the presence of diagonal and super-diagonal blocks with very large elements. These other limiting forms can lead to multiple-frequency-scale perturbed systems. That is, some of the system poles can approach infinity while others remain finite yielding singularly perturbed behavior. Because of the intimate connection between the GHR and the geometric approach to system theory, this work should have theoretical as well as practical applications. That is, it should lead to geometric theories of model reduction which involve both regular and singular perturbations.

[1] V. R. Saksena, J. O'Reilly, and P. V. Kokotovic, "Singular perturbation and time scale methods in control theory: Survey 1976-1983," Automatica, vol. 20, no. 3, pp. 273-293, May 1984.

[2] D. W. Luse and H. K. Khalil, "Frequency Domain Results for Systems with Slow and Fast Dynamics," IEEE Trans. Automat. Contr., vol. AC-30, no. 12, Dec. 1985.

[3] D. K. Lindner, W. R. Perkins, and J. Medanic, "Chained aggregation and three-control-component design--a geometric analysis," Int. J. Contr., vol. 35, pp.621-636, 1982.

[4] H. H. Rosenbrock, State Space and Multivariable Theory. New York: Wiley, 1970.

UNIQUENESS OF THE MINIMAL DYNAMIC COVER AND THE ASSOCIATED SOLUTION TO SYLVESTER'S EQUATION

A. F. ASSAL AND I. B. RHODES
Electrical and Computer Engineering Department
University of California at Santa Barbara
Santa Barbara, Ca 93106

1. INTRODUCTION

In this paper we give necessary and sufficient conditions for the uniqueness of the minimal dynamic cover [1] for a prescribed subspace ***K***. Also given are necessary and sufficient conditions for the uniqueness (modulo a similarity transformation) of the matrix F that satisfies, for a given pair (A',C'), Sylvester's equation TA - FT= GC for some matrix G with appropriate dimension, where the range of T' is a minimal cover for ***K***. These necessary and sufficient conditions are in terms of the obsevability indices of the augmented pair ((C' | K')',A) where the columns of K span the subspace***K***.

For a given pair (C, A) a **dynamic cover** of a prescibed subspace, ***K***, is defined in [1] as any (A',C')-invariant subspace [2] that, together with ***C***, the range of C', contains ***K***. Hence, if τ_{dc} is a cover for ***K*** it satisfies the equations:

$$A'\tau_{dc} \subset \tau_{dc} + C \qquad (1)$$

$$K \subset \tau_{dc} + C \qquad (2)$$

A cover of a given subspace is minimal if there is no cover with smaller dimension among all the covers for of this subspace.

Emre et al [3] gave necessary and sufficient conditions for the uniqueness of the minimal cover of a given subspace. The procedure involved solving a succession of subproblems each identical to the original and having a lower dimension than the previous one. The solution to any subproblem was obtained by solving the next subproblem in the sequence. Emre also showed that the problem of finding a minimal dynamic cover is equivalent to the problem of finding a minimal realization i.e matrices (C, A, B), of a transfer function matrix $G(z) = A^{-1}(z)B(z)$.

2. PRELIMINARIES

1. Consider the augmented system (K_C, A), where K_C=(C' | K')', where the range of K' is the prescribed subspace ***K***, and compute the observability indices [1,4] associated with the augmented system.

2. Let m_i, for i=1,...., m, and n_j, for j=1,..., q, be the observability indices associated, respectively, with the rows of C and the rows of K.

3. Assuming that (C, A) is completely observable then:

$$\sum_{i=1}^{m} m_i + \sum_{j=1}^{q} n_j = n$$

and the dimension of the minimal cover of ***K*** is:

$$\sum_{i=1}^{q} n_i$$

4. A cover for ***K*** satisfies eqns.(1) and (2) which are, as shown in [5], equivalent, respectively, to:

$$TA - FT = GC \qquad \text{(Sylvester's Equation)} \qquad (1a)$$
$$K = MT + NC \qquad (1b)$$

Hence if the columns of T' span a cover of ***K*** then such a T' satisfies the two latter equations for some F, G, M and N.

3. RESULTS

Uniqueness of the Minimal Cover

In this Section, necessary and sufficient conditions for uniqueness of the minimal cover for ***K*** are given in terms of the observability indices associated with the rows of K and C. If the columns of T' span a cover of $\mathcal{K}$, then T satisfies Sylvester's equation:

$$TA - FT = GC \tag{1a}$$

for some F and G. Uniqueness of the cover implies uniqueness of T (modulo some row operations).

Theorem 1

The minimal cover of a given subspace $\mathcal{K}$ is unique iff all observability indices associated with the rows of C are greater than all observability indices associated with the rows of K, where the the columns of K' span the subspace $\mathcal{K}$.

Uniqueness of the F matrix that Satisfies Sylvester's Equation Under specified Conditions

The following theorem gives necessary and sufficient conditions for the uniqueness of the matrix F that satisfies the equation TA - FT = GC, where $\tau_j = R(T_j)'$ spans a minimal cover for some prescribed $\mathcal{K}$.

Theorem 2

The matrix F that satisfies the equation TA - FT = GC, where the columns of T' form a basis for a minimal cover of a prescribed subspace $\mathcal{K}$, is unique (modulo a similarity transformation), iff

i. All m_i, for i =1,..., m, are greater than largest n_j, for j=1,........., q or,

ii. For all m_i, for i =1,...., m, that are less than or equal to the largest n_j, for j=1,..., q; $(c_i)'A^{m_i}$ does not have components along the rows of Q i.e.

$$(c_i)'A^{m_i} = (v_i)'C_a$$

where the columns of Q' span any cover for $\mathcal{K}$ and those of $(Q'|C_a')$ span the whole space, and $C_a = [(M_1)' \ldots..(M_m)']'$ and $(M_i)' = [c_i \ldots\ldots\ldots (A^{m_i-1})' c_i]$, $\forall$ i =1,...., m and c_i is the i^{th} column of C'.

Remarks

1. Both theorems are proven by solving Sylvester's equation and showing that the uniqueness of T (modulo some row operations) in Theorem 1, and that the uniqueness of F (modulo a similarity transformation) in Theorem 2, holds iff the conditions given in the Theorems are satisfied. Complete proofs are given in [6,7].

2. We conclude from Theorems 1 and 2 that uniqueness of the minimal cover implies uniqueness of F (modulo a similarity transformation) that satisfies the equation TA - FT = GC, where the columns of T' span the unique minimal cover. On the other hand the converse is not true. The uniqueness of F (modulo a similarity transformation) of dimension equal to that of the minimal cover for the subspace $\mathcal{K}$ does not imply uniqueness of the minimal cover for $\mathcal{K}$.

REFERENCES

[1] W. M. Wonham and A. S. Morse, "Feedback Invariants of Linear Multivariable systems," Automatica, Vol. 8, No.1, pp. 93-100, Jan. 1972.

[2] W. M. Wonham, "Linear Multivariable Control : a Geometric Approach," (New York: Springer-Verlag), 1979.

[3] E. Emre, L. M. Silverman and K. Glover, "Generalized Dynamic Cover for Linear Systems with Applications to Deterministic Identification and Realization Problems," IEEE Trans. on Automatic Control, Vol. AC-22, No. 1, pp. 26-35, February 1977.

[4] H. Kimura, "Geometric Structure of Observers for Linear Feedback Control Laws," IEEE Trans. on Automatic Control, Vol. AC-22, No. 5, pp. 846 -855, October 1977.

[5] W. M. Wonham, "Dynamic Observers- Geometric Theory," IEEE Trans. on Automatic Control, Vol. AC-15, No. 2, pp. 258-259, April 1970.

[6] A. Assal, "Issues in the Design of Reduced Order Observers," Ph.D. Dissertation, University of California at Santa Barbara,May 1986.

[7] A. Assal and I. B. Rhodes,"Minimum Order Observers- A New Algorithm," to be submitted to the IEEE Trans. on Automatic Control.

EQUIVALENT CHARACTERIZATIONS OF DETECTABILITY AND STABILIZABILITY OF LINEAR TIME-VARYING SYSTEMS

HENG-MING TAI
Department of Electrical Engineering/Computer Science
Texas Tech University
Lubbock, Texas 79409

ABSTRACT

This paper addresses four characterizations of detectability (and stabilizability) for a broad class of linear continuous-time time-varying systems. It is shown that for the class of constant rank systems all these notions of detectability (and stabilizability) are actually equivalent to each other.

1. INTRODUCTION

The dual concepts of detectability and stabilizability for linear time-invariant finite-dimensional systems are of great importance in a variety of control and filtering problems. For instance, these two conditions together are necessary and sufficient for the existence and asymptotic stability of the solutions of the Kalman filter and the Riccati equation. In fact, different definitions of these properties were used to achieve the same goal. However, it is not clear how such various definitions of the concepts of detectability and stabilizability are related to each other. Fortunately, a recent paper by Bittanti and Bolzern [1] elucidates their relationship by proving that four notions of stabilizability of linear periodic systems are actually equivalent.

In this paper, motivated by [1], we consider four characterizations of detectability for a broad class of linear continuous-time time-varying systems. It is shown that for the class of constant rank systems all these notions of detectability are actually equivalent to each other. Dual definitions of the corresponding stabilizability of constant rank systems can be easily obtained. Moreover, their equivalence can be proved by closely following the proof of this paper and by the principle of duality.

2. MAIN RESULTS

Consider a constant rank system [2] $\theta = (F,G,H)_n$ described as

$$\dot{x}(t) = F(t)x(t) + G(t)u(t)$$
$$y(t) = H(t)x(t)$$

where $x(t) \in \mathbf{R}^n$, $u(t) \in \mathbf{R}^m$, $y(t) \in \mathbf{R}^p$ and $F(t)$, $G(t)$, $H(t)$ are matrices of continuous functions of time, sufficiently differentiable and of appropriate dimensions.

DEFINITION 1 (W-detectability). The system θ, or the pair $(F(t),H(t))$ is W-detectable if there exists a bounded continuous matrix $M(t)$ such that the system

$$\dot{x}(t) = [F(t) + M(t)H(t)]x(t)$$

is asymptotically stable.

DEFINITION 2 (K-detectability). The system θ, or the pair $(F(t),H(t))$ is K-detectable if its unobservable part is asymptotically stable.

This definition makes sense in view of the canonical decomposition of constant rank systems [3].

DEFINITION 3 (S-detectability). The system θ is said to be S-detectable if, with $u(t) \equiv 0$, $y(t) \to 0$ as $t \to \infty$ implies $x(t) \to 0$ as $t \to \infty$ for any initial state.

DEFINITION 4 (G-detectability). The system θ is G-detectable if there exist $\alpha,\beta > 0$ and constants a,b with $0 \leq a < 1$, $0 < b < \infty$, such that whenever

$$\|\Phi(t+\alpha,t)\xi\| \geq a\|\xi\|$$

for all $t \in [0,\infty)$ and some ξ, then

$$\xi^T G_O(t,t+\beta)\xi \geq b\xi^T\xi$$

where $G_O(t,t+\beta) = \int_t^{t+\beta}\Phi^T(s,t)H^T(s)H(s)\Phi(s,t)ds$

is the observability Gramian matrix, $\Phi(t,s)$ is the state transition matrix of $\dot{x}(t) = F(t)x(t)$ and $\|\cdot\|$ denotes the Euclidean norm.

REMARK. Notice that the H- and G-stabilizability considered in [1], which are modally characterized in terms of eigenvalues of the monodromy matrix of linear periodic systems, cannot be straightforwardly extended to the time-varying case. Hence we have the alternative definitions stated above. As a matter of fact, S-detectability was originated in [4] for the time-invariant case. G-detectability, which is based on the observability Gramian, was introduced in [5] for discrete-time time-varying systems.

Now we state the main results in the following theorem. Because of space restrictions, proofs are omitted.

THEOREM. W- ,K- ,S- and G-detectability of the system θ are mutually equivalent.

3. CONCLUSION

The equivalence theorem of four characterizations of detectability for constant rank systems has been presented. We believe that this consequence can be used to extend some known control and filtering results to important classes of linear continuous-time time-varying systems.

REFERENCES

[1] S. Bittanti and P. Bolzern, Stabilizability and detectability of linear periodic systems, *Systems and Control Letters* **6** (1985) 141-145.

[2] L.M. Silverman, Realization of linear dynamical systems, IEEE Trans. *Autom. Contr.* **AC-16** (1971) 554-567.

[3] H.-M. Tai and E. Emre, On stabilizability and detectability of linear time-varying systems, in *Proc. of 29th Midwest Symposium on Circuits and Systems*, Lincoln, NE (1986).

[4] L.M. Silverman, Discrete Riccati equations: alternative algorithms, asymptotic properties, and system theory interpretations, in: C.T. Leondes, Ed. *Advanced in Control and Dynamical Systems* (Academic Press, New York, 1976) pp. 313-386.

[5] B.D.O. Anderson and J.B. Moore, Detectability and stabilizability of time-varying discrete-time linear systems, *SIAM J. Control Optim.* **19** (1981) 20-32.

DECENTRALIZED FIXED MODES FOR GENERAL IMPROPER SYSTEMS

T.N. CHANG, E.J. DAVISON
Department of Electrical Engineering
University of Toronto
Toronto, Ontario, Canada M5S 1A4

ABSTRACT

Properties and characterizations of the decentralized fixed modes are obtained for a linear time-invariant system described by:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

where no assumptions are made re the structure of D. In particular, a generalization of known results, re the properties and characterizations of decentralized fixed modes when D has a decentralized structure, are obtained.

1. INTRODUCTION

In the study of large scale systems, the application of centralized control synthesis techniques is limited, and hence this has lead to a study of so-called decentralized control theory, and its related design techniques. An important notion that has arisen from this theory is that of decentralized fixed modes which may be thought of as being an extension of the usual notion of controllability/observability for centralized systems. There has been much development done in this area, e.g. see [1]-[3], [7]-[11].

In these studies, it is typically assumed that the plant to be regulated has the structure:

$$\dot{x} = Ax + \sum_{i=1}^{\nu} B_i u_i$$

$$y_i = C_i x + D_i u_i\,, \qquad i=1,2,\ldots,\nu$$

where the D_i may or may not be present. It is the purpose of this paper to generalize the class of problems which can be solved, by investigating plants which may have the structure:

$$\dot{x} = Ax + \sum_{i=1}^{\nu} B_i u_i$$

$$y_i = C_i x + D_{i1} u_1 + D_{i2} u_2 + \ldots + D_{i\nu} u_\nu\,, \qquad i=1,2,\ldots,\nu$$

In particular, it is desired to characterize the decentralized fixed modes and study their properties, for such a general system. It is to be emphasized that the inclusion of the additional terms D_{ij}, $i \neq j$ is not just a trivial extension of the previous results.

This work has been supported by the Natural Sciences and Engineering Research Council of Canada under grant no. A4396.

2. DEVELOPMENT

The plant is assumed to be represented by a finite-dimensional, linear, time-invariant model:

$$\begin{aligned} \dot{x} &= Ax + B_1 u_1 + B_2 u_2 + \dots + B_\nu u_\nu \\ y_i &= C_i x + D_{i1} u_1 + D_{i2} u_2 + \dots + D_{i\nu} u_\nu , \quad \forall i \in [1,\nu] \end{aligned} \tag{1}$$

where

$$A \in \boldsymbol{R}^{n \times n}$$

$$B_i = (b_{M_{i-1}+1}, b_{M_{i-1}+2}, \dots, b_{M_i}) \in \boldsymbol{R}^{n \times m_i}, \quad i = 1,2,\dots,\nu$$

$$C_i = \begin{pmatrix} c_{R_{i-1}+1} \\ c_{R_{i-1}+2} \\ \vdots \\ c_{R_i} \end{pmatrix} \in \boldsymbol{R}^{r_i \times n}$$

$$D_{ij} = \begin{pmatrix} d_{R_{i-1}+1, M_{j-1}+1} & \cdots & d_{R_{i-1}+1, M_j} \\ \vdots & & \vdots \\ d_{R_i, M_{j-1}+1} & \cdots & d_{R_i, M_j} \end{pmatrix} \in \boldsymbol{R}^{r_i \times m_j}$$

for $\forall i, j \in [1,\nu]$. Define $M_0 = R_0 \equiv 0$ and $M_i = \sum_{j=1}^{i} m_j$, $R_i = \sum_{j=1}^{i} r_j$, $q = \sum_{i=1}^{\nu} m_i r_i$. This notation is required in the development to follow. Finally, denote $m = M_\nu$, $r = R_\nu$, and

$$B = (b_1, b_2, \dots, b_m) \in \boldsymbol{R}^{n \times m}$$

$$C = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_\nu \end{pmatrix} \in \boldsymbol{R}^{r \times n}$$

$$D = (d_1, d_2, \dots, d_m) \in \boldsymbol{R}^{r \times m}, \quad \text{where } d_i = \begin{pmatrix} d_{1i} \\ d_{2i} \\ \vdots \\ d_{ri} \end{pmatrix} \in \boldsymbol{R}^{r}$$

Assume that the following output feedback is to be applied to (1):

$$u = Ky , \quad K \in \boldsymbol{R}^{m \times r} \tag{2}$$

so that the closed loop system is given by

$$\begin{aligned} \dot{x} &= (A + BK(I - DK)^{-1} C)x \\ y &= Cx + Du \end{aligned} \tag{3}$$

provided that $(I - DK)^{-1}$ exists.

The following definitions are now made:

Definition 1: Decentralized Information Flow Constraint

The output feedback (2) is said to have a decentralized information flow constraint imposed on it if $K \in \underline{K}$ where

$$\underline{K} = \{K \in \boldsymbol{R}^{m \times r} \mid k_{ij} \equiv 0, \forall (i,j) \notin \{(i_1,j_1),(i_2,j_2),\ldots,(i_s,j_s)\}, i_k \in [1,m], \\ j_k \in [1,r], k \in [1,s], s \in [1,rm] \wedge \det(I-DK) \neq 0\} \tag{4}$$

Definition 2: Decentralized Fixed Modes

The system (1) with output feedback (2) has a decentralized fixed mode $\lambda \in \mathbb{C}$ with respect to $\underline{K}$ if

$$\lambda \in \bigcap_{\forall K \in \underline{K}} \sigma(A + BK(I-DK)^{-1}C) \tag{5}$$

Definition 3 [6]: Transmission Zeros

Given a system

$$\begin{aligned} \dot{x} &= Ax + Du \\ y &= Cx + Du \end{aligned} \tag{6}$$

where $A \in \boldsymbol{R}^{n \times n}$, $B \in \boldsymbol{R}^{n \times m}$, $C \in \boldsymbol{R}^{r \times n}$, $D \in \boldsymbol{R}^{r \times m}$. The transmission zeros of (6) are defined to be the set of complex numbers which satisfy the following inequality:

$$\rho \begin{pmatrix} A-\lambda I & B \\ C & D \end{pmatrix} < n + \min(r,m) \tag{7}$$

Finally, the following preliminary results are required:

Lemma 1 [4]: Let $M_0 \in \mathbb{C}^{n \times n}$, $b \in \boldsymbol{R}^n$, $c \in \boldsymbol{R}^{1 \times n}$, $\mu \in \boldsymbol{R}$, then $\rho(M_0 + \mu bc) = p < n$, $\forall \mu \in \boldsymbol{R}$ if and only if

$$\rho(M_0, b) = p \quad \text{or} \quad \rho \begin{pmatrix} M_0 \\ c \end{pmatrix} = p \,.$$

Lemma 2 [5]: Given $A \in \mathbb{C}^{n \times n}$, $B \in \mathbb{C}^{n \times n}$, **and** $\det(A + \lambda B) \equiv 0$, $\forall \lambda \in \mathbb{C}$, then $\exists v(\lambda) \neq 0$, such that

$$(A + \lambda B) v(\lambda) = 0$$

Furthermore, degree of λ in $v(\lambda) \leq \operatorname{rank} B$.

3. MAIN RESULTS

The following results are now obtained:

Theorem 1: (Determinantal Characterization of Decentralized Fixed Modes)

For the plant (1), let $m = r$, $\lambda \in \sigma(A)$ and $K \triangleq \operatorname{diag}(k_1, k_2, \ldots, k_r) \in \boldsymbol{R}^{r \times r}$, then

$$\det(A - \lambda I + BK(I-DK)^{-1}C) \equiv 0, \quad \forall k_1, k_2, \ldots, k_\nu \in \boldsymbol{R}$$

if and only if the following conditions all hold:

$$\det\begin{bmatrix} A-\lambda I & b_{i_1} & b_{i_2} & \cdots & b_{i_t} \\ c_{i_1} & d_{i_1 i_1} & d_{i_1 i_2} & \cdots & d_{i_1 i_t} \\ \vdots & \vdots & & & \vdots \\ c_{i_t} & d_{i_t i_1} & & \cdots & d_{i_t i_t} \end{bmatrix} \equiv 0$$

where $i_t \in [1,r]$, $t \in [1,r]$, and $i_1, i_2, \ldots i_t$ are disjoint.

Δ

Proof: All proofs are given in the appendix.

Remarks: The above results relate the decentralized fixed mode condition to a series of determinantal conditions which are independent of K. The next step is to relate a general, centralized feedback configuration to a decentralized one, such that the two systems have the same spectrum and fixed modes. This is provided by the next theorem.

Theorem 2: (Decentralized Expansion of Arbitrary Systems)

For the plant (1), let $u = Ky$ be a centralized output feedback; then there exists $\tilde{B} \in \mathbf{R}^{n \times rm}$, $\tilde{C} \in \mathbf{R}^{rm \times n}$, $\tilde{D} \in \mathbf{R}^{rm \times rm}$, and $\tilde{K} \in \mathbf{R}^{rm \times rm}$ such that $\tilde{K} = \mathrm{diag}(k_{11}, k_{12}, \ldots, k_{1r}, k_{21}, k_{22}, \ldots, k_{2r}, \ldots, k_{m1}, \ldots, k_{mr})$ and

$$A - \lambda I + BK(I - DK)^{-1}C \equiv A - \lambda I + \tilde{B}\tilde{K}(I - \tilde{D}\tilde{K})^{-1}\tilde{C}\ , \quad \forall \lambda \in \sigma(A)$$

In particular,

$$\tilde{B} = [\underbrace{b_1 \ \cdots \ b_1}_{r\text{-columns}} \mid \underbrace{b_2 \ \cdots \ b_2}_{r\text{-columns}} \mid \ldots \mid \underbrace{b_m \ \cdots \ b_m}_{r\text{-columns}}]$$

$$\tilde{C} = \begin{bmatrix} C \\ C \\ \vdots \\ C \end{bmatrix} \left.\vphantom{\begin{bmatrix} C \\ C \\ \vdots \\ C \end{bmatrix}}\right\} \begin{matrix}\text{repeated} \\ m\text{ times}\end{matrix}, \text{ and } \tilde{D} = \begin{bmatrix} \underbrace{\begin{matrix} d_1 & \cdots & d_1 \\ \vdots & & \\ d_1 & \cdots & d_1 \end{matrix}}_{r\text{-columns}} & \underbrace{\begin{matrix} d_2 & \cdots & d_2 \\ & \vdots & \\ d_2 & \cdots & d_2 \end{matrix}}_{r\text{-columns}} & \cdots & \underbrace{\begin{matrix} d_m & \cdots & d_m \\ & \vdots & \\ d_m & \cdots & d_m \end{matrix}}_{r\text{-columns}} \end{bmatrix} \left.\vphantom{\begin{bmatrix} d \\ d \\ d \end{bmatrix}}\right\} \begin{matrix}\text{repeated} \\ rm\text{ times}\end{matrix}$$

constitute an admissible choice.

Δ

The decentralized fixed modes of (1) with respect to $\underline{K}$ are now characterized by their corresponding transmission zero conditions, which is a generalization of [2]:

Theorem 3: (Transmission Zero Characterization of Decentralized Fixed modes)

For the plant (1), $\lambda \in \sigma(A)$ is a decentralized fixed mode with respect to $\underline{K}$ if and only if λ is a transmission zero of all of the following systems:

$$\left\{\begin{pmatrix} C_{j_{k1}} \\ C_{j_{k2}} \\ \vdots \\ C_{j_{kt}} \end{pmatrix}, A\,(b_{i_{k1}}, b_{i_{k2}}, \ldots, b_{i_{kt}}), \begin{pmatrix} d_{j_{k1}i_{k1}} & , & d_{j_{k1}i_{k2}} & , \cdots, & d_{j_{k1}i_{kt}} \\ d_{j_{k2}i_{k1}} & , & & , \cdots, & \\ \vdots & & & & \vdots \\ d_{j_{kt}i_{kt}} & , & & , \cdots, & d_{j_{kt}i_{kt}} \end{pmatrix}\right\}$$

where

$$k_{i+1} \in [k_i+1, s-t+1+i]$$

$$i = 0,1,\ldots,s-1\,, \quad t = 1,2,\ldots,\min(m,r)$$

$$k_0 \equiv 0$$

and the i_k, j_k's are given in definition 1.

Δ

Remarks: Theorem 3 completely captures the decentralized fixed mode conditions by considering all possible feedback information flow constraints. Therefore, the following standard decentralized fixed mode information flow constraint:

$$K = \text{block diag}(K_1, K_2, \ldots, K_\nu) \tag{8}$$

where $K_i \in \boldsymbol{R}^{m_i \times r_i}$, may be easily handled by applying theorem 3 to the following set of decentralized information flow constraint:

$$\underline{K} = \{K \in \boldsymbol{R}^{m\times r} \mid k_{ij} \equiv 0, \forall (i,j) \notin \{(1,M_1)\times(1,R_1) \cup (M_1+1,M_2)\times(R_1+1,R_2) \cup \cdots$$
$$\cup (M_{\nu-1}+1,M_\nu)\times(R_{\nu-1}+1,R_\nu) \wedge \det(I-DK) \neq 0\}\}$$

However, an alternative characterization similar to that obtained in [3] can be also derived. This is given by the next theorem.

Theorem 4: (Alternative Characterization of Decentralized Fixed Modes)

Let $\lambda \in \sigma(A)$, then λ is decentralized fixed mode of (1) with respect to information flow restrictions (8) if and only if any one of the following conditions hold:

1) $\rho\begin{pmatrix} A-\lambda I \\ C_1 \\ \vdots \\ C_\nu \end{pmatrix} < n$

2) $\rho(A-\lambda I, B_1, \ldots, B_\nu) < n$

3) $\rho\begin{pmatrix} A-\lambda I & B_{i_1} \\ C_{i_2} & D_{i_2,i_1} \\ \vdots & \vdots \\ C_{i_\nu} & D_{i_\nu,i_1} \end{pmatrix} < n$

for some $i_k \in [1,2,\ldots,\nu]$, $k=1,2,\ldots,\nu$ such that $\{i_1, i_2, \ldots, i_\nu\} = \{1,2,\ldots,\nu\}$.

4) $\rho\begin{pmatrix} A-\lambda I & B_{i_1} & B_{i_2} \\ C_{i_3} & D_{i_3 i_1} & D_{i_3 i_2} \\ \vdots & \vdots & \vdots \\ C_{i_\nu} & D_{i_\nu i_1} & D_{i_\nu i_2} \end{pmatrix} < n$

for some $i_k \in [1,2,...,\nu]$, $k=1,2,...,\nu$ such that $\{i_1,i_2, \ldots, i_\nu\}=\{1,2,...,\nu\}$.

$\vdots$

$$\nu+1)\rho\begin{pmatrix} A-\lambda I & B_{i_1} & B_{i_2} & \cdots & B_{i_{\nu-1}} \\ C_{i_\nu} & D_{i_\nu i_1} & D_{i_\nu i_2} & \cdots & D_{i_\nu i_{\nu-1}} \end{pmatrix} < n$$

for some $i_k \in [1,2,...,\nu]$, $k=1,2,...,\nu$ such that $\{i_1,i_2,...,i_\nu\}=\{1,2,...,\nu\}$.

The following example illustrates the application of theorem 4. Assume that $\nu=2$ and $\lambda \in \sigma(A)$, then λ is a decentralized fixed mode with respect to (8) if and only if any one of the following four conditions hold:

1) $\rho(A-\lambda I, B_1, B_2) < n$

2) $\rho\begin{pmatrix} A-\lambda I \\ C_1 \\ C_2 \end{pmatrix} < n$

3) $\rho\begin{pmatrix} A-\lambda I & B_1 \\ C_2 & D_{21} \end{pmatrix} < n$

4) $\rho\begin{pmatrix} A-\lambda I & B_2 \\ C_1 & D_{12} \end{pmatrix} < n$

It should be noted that the D_{ii}'s, do not play any role in the characterization of decentralized fixed modes. This observation is perhaps not surprising, since the available local information enables one to "remove" the D_{ii} terms from a feedback controller.

The structural characterization of decentralized fixed modes is now completely captured by theorems 3 and 4. The numerical computation of the decentralized fixed modes can also be obtained by generalizing the results in [7] to the present system (1).

The following lemma is essential in the derivation of algorithm 1.

Lemma 3: Given $A \in \mathbf{R}^{n\times n}$, $B \in \mathbf{R}^{n\times m}$, $C \in \mathbf{R}^{r\times n}$, $D \in \mathbf{R}^{r\times m}$, then the class of matrices $K \in \underline{K}$ which does not result in the decentralized fixed modes of (C,A,B,D) being equal to those eigenvalues of $A+BK(I-DK)^{-1}C$, which are in common with the eigenvalues of A, is either empty or lies on a hypersurface [7] in the parameter space of K.

The set of decentralized fixed modes can now be computed as follows:

Algorithm 1: (Determination of Decentralized Fixed Modes)

1) Find the eigenvalues of A.

2) Select an arbitrary feedback matrix $K \in \underline{K}$ (e.g. by using a pseudo-random number generator) so that

$$||A|| \simeq ||BK(I-DK)^{-1}C||$$

3) Find the eigenvalues of $(A+BK(I-DK)^{-1}C)$.

4) Then the set of decentralized fixed modes with respect to $\underline{K}$ is contained in $\sigma(A) \cap \sigma(A+BK(I-DK)^{-1}C)$. In fact, the set of decentralized fixed modes is equal to $\sigma(A) \cap \sigma(A+BK(I-DK)^{-1}C)$ for "almost all" $K \in \underline{K}$.

5) Repeat steps 2-4) using a different K if in doubt.

Discussion: An important application of decentralized fixed modes for system of the type described by (1), is that they characterize the solvability of the decentralized stabilization problem for systems described by (1). Necessary and sufficient conditions for there to exist a linear time invariant decentralized controller to stabilize (1), are given in a companion paper [12], where it is shown that the conditions correspond to the requirement that all decentralized fixed modes should be stable.

4. EXAMPLES

Consider the following "large-scale system":

$$\dot{x} = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} x + \begin{pmatrix} 0 \\ 0 \end{pmatrix} u_1 + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u_2$$

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} x + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} u \tag{9}$$

subject to the following decentralized information flow constraint:

$$k_{ij} = 0 , \quad \forall i \neq j , \quad i,j = 1,2$$

i.e.

$$K = \mathrm{diag}(k_1,k_2) \in \boldsymbol{R}^{2\times 2}$$

The following results are obtained:

(A) Applying theorem 3, $\lambda = 2 \in \sigma(A)$ is a decentralized fixed mode if and only if the following conditions all hold:

i) $\rho \begin{pmatrix} A-2I & b_1 \\ c_1 & d_{11} \end{pmatrix} < n+1$

ii) $\rho \begin{pmatrix} A-2I & b_2 \\ c_2 & d_{22} \end{pmatrix} < n+1$

iii) $\rho \begin{pmatrix} A-2I & b_1 & b_2 \\ c_1 & d_{11} & d_{12} \\ c_2 & d_{21} & d_{22} \end{pmatrix} < n+2$

Now since:

i) $\rho \begin{pmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{pmatrix} = 2 < n+1 = 3$

ii) $\rho \begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 1 \\ 0 & 1 & 0 \end{pmatrix} = 2 < n+1 = 3$

iii) $\rho \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix} = 4 = n+2$

it is concluded that $\lambda=2$ is not a decentralized fixed mode, since condition

iii) is violated.

(B) Alternatively, theorem 4 may be applied, i.e. $\lambda=2$ is a decentralized fixed mode if any one of the following conditions hold:

i) $\rho(A-2I, b_1, b_2) < n$

ii) $\rho\begin{pmatrix} A-2I \\ c_1 \\ c_2 \end{pmatrix} < n$

iii) $\rho\begin{pmatrix} A-2I & b_1 \\ c_2 & d_{21} \end{pmatrix} < n$

iv) $\rho\begin{pmatrix} A-2I & b_2 \\ c_1 & d_{12} \end{pmatrix} < n$

Now since:

i) $\rho\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 \end{pmatrix} = 2 = n$

ii) $\rho\begin{pmatrix} 0 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} = 2 = n$

iii) $\rho\begin{pmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 1 \end{pmatrix} = 2 = n$

iv) $\rho\begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 1 \\ 1 & 0 & 0 \end{pmatrix} = 3 > n$

it is again verified that $\lambda=2$ is not a decentralized fixed mode. It is of interest to note that if d_{21} is set to zero, then $\lambda=2$ now becomes a decentralized fixed mode.

(C) Numerically, algorithm 1 may be applied to determine if the system (9) has any decentralized fixed modes as follows:

1) computer $\sigma(A)$; then $\sigma(A)=\{2,1\}$

2) arbitrarily select $K=\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$; then

$$\sigma(A+BK(I-DK)^{-1}C)=\left\{\frac{3\pm j\sqrt{3}}{2}\right\}$$

3) since $\sigma(A)\cap\sigma(A+BK(I-DK)^{-1}C)=\phi$, it is concluded that the system has no decentralized fixed modes.

5. CONCLUSIONS

In this paper, a complete characterization of decentralized fixed modes of a linear, finite-dimensional, time-invariant system has been derived. The fixed modes produced by any decentralized information flow restriction can be related to a series of transmission zero conditions on the subsystems. Alternatively, a set of rank conditions on the "augmented subsystems" is also provided. Although the two characterizations are equivalent, their linkage is not obvious. The D term, which results from a state-space realization of improper, rational transfer matrices, has been shown to play an essential role in determining the fixed modes of a decentralized system, which is not the case in the conventional, centralized multivariable problem. Finally, an algorithm is provided to compute the decentralized fixed modes numerically.

REFERENCES

[1] Wang, S.H., Davison, E.J., "On the stabilization of decentralized control systems", *IEEE Trans. on Auto. Control*, vol. AC-18, no. 5, Oct. 1973, pp. 473-478.

[2] Davison, E.J., Wang, S.H., "A characterization of decentralized fixed modes in terms of transmission zeros", *IEEE Trans. on Auto. Control*, vol. AC-30, no. 1, Jan. 1985, pp. 81-82.

[3] Anderson, B.D.O., Clements, D.J., "Algebraic characterizations of fixed modes in decentralized systems", *Automatica*, vol. 17, no. 5, 1981, pp. 703-712.

[4] Anderson, B.D.O., Hong, H.M., "Structural controllability and matrix nets", *Int. J. Control*, vol. 35, no. 3, 1982, pp. 397-416.

[5] Gantmacher, F.R., *The Theory of Matrices*, vol. II, Chelsea, 1960.

[6] Davison, E.J., Wang, S.H., "Properties and calculations of transmission zeros of linear multivariable systems", *Control Systems Design by Pole-Zero Assignment*, Academic Press, 1977.

[7] Davison, E.J., "Decentralized stabilization and regulation in large multivariable systems", in *Directions in Large Scale Systems*, (editors: Ho, Y.C., and Mitter, S.K.), Plenum Publishing Corp., 1976, pp. 303-323.

[8] Davison, E.J., Ozguner, U., "Characterizations of decentralized fixed modes for interconnected systems", *Automatica*, vol. 19, no. 2, March 1983, pp. 169-182.

[9] IEEE, Special Issue on Large-Scale Networks and Systems, *IEEE Trans. Circuits and Systems*, 23, 1976.

[10] IEEE, Special Issue on Decentralized Control, *IEEE Trans. on Automatic Control*, 23, 1978.

[11] Chang, T.N., Davison, E.J., "Decentralized Pole Assignment for General Improper Systems", University of Toronto, Systems Control Group Report No. 8605, Feb. 1986.

APPENDIX

The appendix is omitted due to page limitation. It may be found in [11].

THE COMPLETE SET OF STABILIZING CONTROLLERS FOR 2-D MULTIVARIABLE SYSTEMS

V. Raman
Department of Electrical Engineering
Florida International University
Miami, Fla. 33199

R. Liu
Department of Electrical Engineering
University of Notre Dame
Notre Dame, In. 46556

ABSTRACT

The characterization of controllers for the design of 2-D multivariable feedback systems is considered here, using the factorization approach. After defining a generalized stability criterion appropriate to 2-D applications, the key coprime factorization problem for 2-D plant is solved. The constructive solution is valid for the various possible 2-D causality conditions. Thus the set of all controllers stabilizing the 2-D feedback system is obtained.

I. INTRODUCTION

This work deals with the design of two-dimensional (2-D) multivariable feedback systems, more specifically, it is sought to obtain desirable feedback system properties which are described by a generalized stability criterion. The goal, then, is to characterize all 2-D controllers which achieve this objective. The formulation of the problem is based on the axiomatic factorization approach [1]-[5], which tackles this problem in a ring-theoretic setting in order to allow applicability of the theory to a wide class of linear systems, including 2-D systems. It has been shown therein that the complete set of generalized-stabilizing controllers for a given plant can indeed be explicitly characterized, provided a crucial equation, termed the coprime condition, can be solved in the ring of interest. This was more complete and generalized version of the result of [6] for 1-D multivariable systems.

The focus of this work is: choosing an appropriately general 2-D generalized-stability criterion; determining the conditions on the plant for stabilizability of the system; solution of the coprime condition for 2-D multivariable systems under differing causality constraints, thus obtaining the complete set of stabilizing controllers. For 2-D scalar system without causality constraints a solution was shown in [7]. Systems treated as 2-D systems include delay, multipass, partial-differential systems, etc..

II. PROBLEM FORMULATION

The standard unity-feedback configuration of Figure 1 is used, where the plant P and controller C are 2-D multivariable systems.

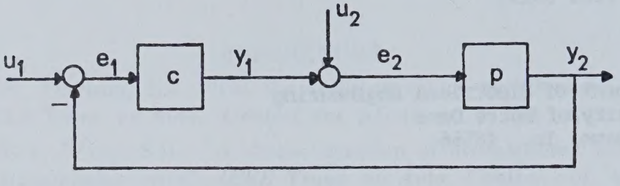


Figure 1

The systems we are concerned with are described by 2-D matrices of rational transfer functions in the variables s and w, with real coefficients. In 2-D systems, various causality conditions are possible. Details on this are relegated to Section IV; the term "proper" 2-D transfer function will be used in this section with the understanding that it be associated with an appropriate causality condition. Definitions and notations now follow.

Let G denote the (commutative) ring of proper, real-coefficient rational functions in the variables s and w. G represents the set of scalar systems. A generalized stability criterion is now formulated in an accepted fashion. Let $\Omega \subset C\times C$ denote an "undesirable" region of the (s,w) hyperplane. Then let H be a subring of G defined by $H \triangleq \{g \text{ in } G \mid$ the denominator polynomial of g has no zeros in $\Omega\}$. Elements of H will be referred to as H-stable. The usual BIBO stability conditions for 2-D systems admit such a characterization; by appropriate choice of Ω, stronger "desirable" properties can be incorporated into H-stability. Adopting the notation M(S) for the set of all matrices over a commutative ring S, M(G) and M(H) represent the set of multivariable systems and H-stable systems, respectively. M_I will denote elements of M(H) with inverses in M(G).

As in the factorization approach, [1]-[5], the feedback design objective is formulated as: given P in M(G), obtain C in M(G) which assures internal H-stability of the feedback system (internal H-stability assures that all transfer functions in the feedback system are H-stable; the exact characterization can be found in the references and is not given here). The results are based on coprime factorizations described below [1]-[3].

$P = ND^{-1}$, N and D in M(H), is said to be a right-coprime factorization (r.c.f.) of P in M(H) if there exist U and V in M(H) satisfying the right coprime condition

$$UN + VD = I \tag{1}$$

Left coprime factorizations (l.c.f.) and coprimeness are defined analogously.

The following theorem gives the key H-stabilization result from the literature.

Theorem O [1]-[3] : Let P in M(G) have a r.c.f. and l.c.f. in M(H), $P = ND^{-1} = D_L^{-1}N_L$, with some U, V in M(H) satisfying (1). Then, C in M(G) internally H-stabilizes the feedback system if and only if

$$C = (V + TN_L)^{-1} (U - TD_L) \tag{2}$$

for some T in M(H) such that $(V+TN_L)$ is in M_I.

First, Theorem O establishes that the existence of coprime factorizations of P in M(H) allows the (theoretical) characterization of all H-stabilizing controllers. In fact, it is shown in [8] that, for 2-D systems, H-stabilization is not possible if coprime factorizations of P do not exist. Thus the controller characterization of Theorem O is truly complete. Second, the parametrization of controllers then requires obtaining a r.c.f. and l.c.f. of P and a solution to the coprime condition (1). This is always possible in 1-D systems. However, in 2-D the presence, in a transfer function, of second-kind singularities (non-cancellable zeros of numerator and denominator polynomials) poses a limitation. The following section deals with this question.

III. 2-D FACTORIZATIONS

Before presenting some useful results towards solving (1), some 2-D notions are outlined (refer [9] - [12] for details). A notational point: the subscript 'p' will signify that the subscripted matrix is a polynomial matrix.

Let

$$P(s,w) = N_p(s,w)\, D_p^{-1}(s,w) \tag{3}$$

be a polynomial matrix right factorization which is minimal, as defined below. Letting $A_p \triangleq [D_p^T\, N_p^T]^T$, $P = N_p D_p^{-1}$ is called a minimal factorization if A_p is a irreducible polynomial matrix, that is, A_p has no non-trivial (non-unimodular) right (polynomial matrix) factor. Furthermore common zeros of the largest order minors of $A_p(s,w)$ will be called the zeros of A_p or the common zeros of $N_p(s,w)$ and $D_p(s,w)$.

Addressing the coprime condition (1), it is readily shown [7] that (1) does not have a solution unless N_p and D_p are Ω-zero-right-coprime (Ω-z.r.c.), that is, they have no common zeros in Ω (extending the usual definition of zero-coprimeness, e.g. [11]. Furthermore, it can be shown by counter-example that even if they are Ω-zero-coprime, (1) is not solvable for arbitrary choice of Ω [7]. In the following, therefore, a structure is imposed on Ω .

Separability Assumption:

$$\Omega = \Omega_s \times \Omega_w$$

where $\Omega_s \subset C$, $\Omega_w \subset C$.

That is, Ω is the cartesian product of regions in the s-plane and w-plane, respectively. (From the viewpoint of real-coefficient solutions for the controller, Ω and Ω are assumed closed under conjugation). Note that the BIBO stability criteria for common 2-D applications, e.g.: 2-D filters: $\Omega = \{(s,w) \mid |s| \geq 1, |w| \geq 1\}$; delay-differential equations: $\Omega = \{(s,w) \mid Re(s) \geq 0, |w| \quad 1\}$, etc., satisfy this structure. The assumption does not appear to restrictive, but it does facilitate the solution of the coprime condition.

The main result in this work is in Theorem 2 of Section IV. However, the following Theorem 1 is not only an essential step in showing Theorem 2, but is also of independent interest.

Theorem 1 [8]: For P in M(G), let $P = N_p D_p^{-1}$ be a minimal polynomial matrix factorization of P. Then, N_p and D_p are Ω-zero-right-coprime if and only if there exist U and V in M(H) such that

$$U N + V D = I \tag{3}$$

The 'if' part of the theorem is intuitively obvious and easily proved. The steps are: write (3) as $[U \quad V] A_p = I$; clear denominators (polynomials with no zeros in) of [U V] by premultiplication; by the Binet-Cauchy theorem, the zeros of A_p cannot lie in Ω.

The 'only if' part of the Theorem I is constructive. It relies on a key lemma (described later) and is essential to Theorem 2 in Section IV, but is not detailed here due to space limitations.

Consideration of Theorem 1 for the special case $\Omega = C \times C$ is interesting, since M(H) then became the set of 2-D polynomial matrices. (3) is then the well known Hilbert Identity and Theorem 1 shows the 2-D matrix Hilbert Nullstellensatz - namely that zero coprimeness of the polynomial matrices N_p and D_p is equivalent to solvability of (3). A scalar versionof this construction can be found in [7]. Solutions to this polynomial matrix problem differing from the construction of Theorem 1 can also be found in [9], [11], [12].

The key to the construction of U and V of Theorem 1 lies in combining the separability of Ω with the following lemma.

Separation Lemma: For $A_p(s,w)$ (as defined earlier) an irreducible polynomial matrix, let the enumerated distinct values of w corresponding to the zeros of $A_p(s,w)$ be w_i, $i \in K = \{1,2,..,k\}$. Then, for any $K_1 \subset K$, a polynomial matrix $B_p(s,w)$ can be constructed such that

$$B_p(s,w)\, A_p(s,w) = R_1(w)\, R_2(s) \tag{4}$$

where the polynomial matrices R_1 and R_2 are diagonal, square and have the property that

$\det(R_1(w^\circ)) = 0$ iff $w^\circ = w_i$ for some $i \in K_1$, and

$\det(R_2(s^\circ)) = 0$ iff (s°, w_i) is a zero of $A_p(s,w)$ for some

$$i \in K - K_1 .$$

The constructive proof of this lemma gives explicit B_p, R_1 and R_2 for each choice of K_1. The proof can be found in [8], but a brief outline follows: for irreducible $A_p(s,w)$, a polynomial matrix $B_1(s,w)$ can be found such that

$$B_1(s,w)\, A_p(s,w) = R(w)$$

where R(w) is a square diagonal polynomial matrix (refer [9], [10] for methods). Clearly every w_i, $i \in K$, is a zero of $\det(R(w))$. The subsequent steps involve "elimination" of factors of the form $(w-w_i)$, $i \in K-K_1$ from the right hand side. Consider w_i, $i \in K-K_1$. The first step involves division of $A_p(s,w)$ by $(w-w_i)$ to give a remainder polynomial matrix in s, which has the property it has a zero at s° if and only if (s°, w_i) is a zero of $A_p(s,w)$. Similar procedures are repeated for the other w_i, $i \in K-K_1$, and algebraic manipulations are used to show the final result (4).

Clearly, a symmetric result holds for the distinct values of s corresponding to zeros of $A_p(s,w)$.

In proving Theorem 1, this lemma is particularly useful in conjunction with the separability of Ω that is $\Omega = \Omega_s \times \Omega_w$. The zero-coprimeness of $N_p(s,w)$ and $D_p(s,w)$ shows that $A_p(s,w)$ has no zeros in Ω; hence for any zero (s°, w°) of $A_p(s,w)$, if $s^\circ \in \Omega_s$, then $w^\circ \notin \Omega_w$ and similarly if $w^\circ \in \Omega_w$, then $s^\circ \notin \Omega_s$. Consequently, the lemma can be applied to $A_p(s,w)$ by choosing K_1 such that w_i, $i \in K-K_1$, are those values of w (among the zeros of A_p) which lie in Ω_w, hence the associated values of s (in the zeros of A_p) cannot lie in Ω_s. Hence in (4), $R_1(w)$ and $R_2(s)$ have no zeros in Ω_w and Ω_s respectively, and pre-multiplication of (4) by $(R_1 R_2)^{-1}$ gives (3).

The subsequent section completes the compensator construction problem by addressing the "properness" constraint in G and H.

IV PROPER COPRIME FACTORIZATION

To obtain a complete solution to the H-stabilization problem, the issue of 2-D causality or properness of the systems in G and H has to be considered. Some notions in that regard are given here; [13], [8] may be used for details.

Since 2-D systems may be considered to propagate in two directions, several different causality constraints are possible. In terms of "properness" of the 2-D transfer function, this means that properness in the variable s alone, or w alone, or in both s and w, may be required. In the following , we consider (s,w) - properness (properness in s and w), since this is the most restrictive, and the other two are special cases of this.

First, a 2-D polynomial $d_p(s,w)$ is called acceptable [13] if it contains a monomial $d_{oo}\, s^j w^l$ where $j = \deg_s(d_p)$, $l = \deg_w(d_p)$, in other words, the highest degree terms in s and w occur in association. Then, a transfer function $n_p(s,w)/d_p(s,w)$ is called (s,w) - proper if : (i) $d_p(s,w)$ is acceptable, (ii) $\deg_s(d_p) \geq \deg_s(n_p)$ and (iii) $\deg_w(d_p) \geq \deg_w(n_p)$. Note that elimination of (i) and one of (ii) or (iii) will give w-properness or s-properness respectively.

Thus, M(G) is the set of matrices over (s,w)-proper transfer functions, and M(H) is the H-stable subset of M(G).

In a fashion analagous to 1-D multivariable systems, the idea of 2-D column- (or row-) reduced forms [13] are now useful in obtaining simplified characterizations of 2-D transfer matrices. We use a particular column-reduced form here, explained in the following.

$D_p(s,w)$ is column reduced over s and w (more accurately, over the ring of 2-D polynomials in s and w) if $\det(D_p(s,w))$ contains a monomial $d_{oo} s^j w^l$, where j and l are the sums of the column degrees of $D_p(s,w)$ in s and w, respectively.

For the minimal polynomial factorization, $P(s,w) = N_p(s,w)D_p^{-1}(s,w)$, $D_p(s,w)$ is henceforth assumed to be column reduced. Now consider $A_p(s,w)$. Writing

$$A_p(s,w) = A_1(w) \begin{bmatrix} s^{c_1} & & & \\ & s^{c_2} & & \\ & & \ddots & \\ & & & s^{c_m} \end{bmatrix} + \text{lower degree terms in } s$$

$A_1(w)$ denotes the leading column-coefficient matrix of A_p over s. Similarly, let $A_2(s)$ denote the leading column coefficient matrix of A_p over w.

Finally, before presenting Theorem 2, an assumption is required on Ω: assume Ω_s contains a real number k_s and Ω_w contains a real number k_w. The relevance of this is explained later.

Theorem 2: For P in $M(G)$ with the minimal polynomial factorization $P = N_p D_p^{-1}$, with D_p column-reduced, let $A_1(w)$ and $A_2(s)$ have no zeros in Ω_w and Ω_s respectively. Then, N_p and D_p are Ω-zero-right-coprime if and only if there exists a right coprime factorization of P in $M(H)$, $P = N\,D^{-1}$, satisfying

$$UN \quad + \quad VD \quad = I \tag{5}$$

with some U and V in $M(H)$.

The proof of this theorem is constructive, and is equally useful in obtaining r.c.f.s. and l.c.f.s. of P, $P = N\,D^{-1} = D_L^{-1} N_L$, and the solution (U,V) for (5). Before outlining the proof, the following corollary states the complete H-stabilization result by combining Theorems 0 and 2.

Corollary: For $P = N_p D_p^{-1}$ under the conditions of Theorem 2, the feedback system is H-stabilizable by C in $M(G)$ only if N_p and D_p are Ω-z.r.c. Conversely, if N_p and D_p are Ω-z.r.c., equation (5) is solvable and C in $M(G)$ H-stabilizes the system if and only if C is of the form

$$C = (V + TN_L)^{-1}\,(U - TD_L) \tag{6}$$

for some T in $M(H)$ such that $(V + TN_L)$ in M_I.

This essentially provides a complete constructive characterization of the set of H-stabilizing controllers for 2-D multivariable systems.

The construction of Theorem 1 uses a 2-D version of the Λ-generalized polynomials [14] used in 1-D systems, as follows. Let

$$x \triangleq \frac{1}{s-k_s}, \qquad y \triangleq \frac{1}{w-k_w}$$

where real numbers k_s in $\Omega_s^{\,c}$ and k_w in $\Omega_w^{\,c}$ exist by assumption. The transformation of variables from (s,w) to (x,y) is used in $P(s,w)$, mapping Ω_s and Ω_w appropriately to regions in the x- and w- planes respectively, and use of Theorem 1 gives a polynomial matrix solution in the variables x and y. The inverse mapping to (s,w) now gives a solution in $M(H)$.

The assumptions of column-reduced form of $D_p(s,w)$, and on the zeros of $A_1(w)$ and $A_2(s)$ are not essential to solving the coprime factorization, but their absence leads to more elaborate characterizations which serve to obscure the nature of the solution. Note that the conditions on $A_1(w)$ and $A_2(s)$ are akin to "formal stability" conditions used in designing controllers for delay-systems.

When the properness condition is less restrictive, the assumptions on D_p and $A_1(w)$, $A_2(s)$ are relaxed, and the construction is further simplified.

The work closest in spirit to the above is in [9], where the coprime condition is tackled via a method based on [11]. The results shown in our work differ in methodology, and are of wider applicability :this construction is applicable to any, not necessarily strictly proper plant, and furthermore, the possibility of internal instability due to unstable cancellations [15] is absent.

V CONCLUSIONS

This work focuses on applying the abstract results of the factorization approach to two-dimensional multivariable systems, applicable to such as delay systems, partial-differential systems, etc. As such, a constructive method has been developed for characterizing all controllers which can cause the system to (internally) satisfy design criteria represented by a generalized-stability condition.

REFERENCES

[1] Desoer, C.A., R. Liu, J. Murray and R. Saeks. "Feedback Systems Design: The Fractional Representation Approach to Analysis and Synthesis, "IEEE Trans. Automatic Control, Vol. AC-25, pp 399-412, 1980.

[2] Liu, R. and C.H. Sung, "Linear Feedback System Design", Circuits, Systems and Signal Processing, Vol. 2, No. 1, pp 35-44, Birkhauser, 1982.

[3] Vidyasagar, M.H. Schneider and B. Francis, "Algebraic and Topological Aspects of Feedback Stabilization, "IEEE Trans. Automatic Control, Vol. AC-25, pp. 9-412, 1980.

[4] Vidyasagar, M., Control System Synthesis : A Factorization Approach M.I.T. Press, 1984.

[5] Raman, V.R. and R. Liu, "A necessary and Sufficient Condition for Feedback Stabilization in a Factorial Ring, IEEE Trans. Automatic Control. Vol AC-29, pp 941-943, 1984.

[6] Youla, D.C., J.J. Bongiorno and H.A. Jabr, "Modern Wiener-Hopf Design of Optimal Controllers - Part I & II", IEEE Trans. Automatic Control, Vol. AC-21, pp 3-15 and pp 338, 1976.

[7] V.R. Raman and R.W. Liu, "A Constructive Algorithm for the Complete Set of Compensator for 2-D Feedback System Desing", IEEE Trans. Aut. Contr. Vol. AC 31, pp 166-179, 1986.

[8] V.R. Raman, "Factorization Approach in Control: Coprime Conditions

and their Solution". Ph.D Dissertation, University of Notre Dame, 1985.

[9] Levy, B.C. "Algebraic Structures for Two-Dimensional Systems" Ph. D Dissertation, Dept. of Elec. Eng., Stanford University, June 1981.

[10] Morf, M., B.C. Levy and S.Y. Kung, "New Results in 2-D Systems Theory, Part I : 2-D Polynomial Matrices, Factorization and Coprimeness, "IEEE Proceedings, Vol. 65, No. 6, pp 861-872, 1977.

[11] Youla, D. C., and G. Gnavi, "Notes on n-Dimensional System Theory, IEEE Trans. Circuits and Systems, Vol CAS 26, pp 105-111, 1979.

[12] Bose, N. K.Multidimensional Systems Theory: Progress, Directions and Open Problems in Multidimensional Systems Hingham, M.A. : Kluwer Academic Publishers, 1985.

[13] Chiasson, J. N., S.D. Brierly, S.H. Zak and E.B. Lee, "On Proper 2-D Transfer Functions, Int. J. of Control, Vol. 35 No. 1 pp 159-174, 1982.

[14] Pernebo, L., "An Algebraic Theory for the Design of Controllers for Linear Multivariable Feedback Systems", IEEE Trans. Automatic Control Vol. AC-21, pp 171-194, 1981.

[15] Zhang, S.H, "Comments on 2-D Exact Model-Matching Problem", Sebek, M. "Author's Reply", IEEE Trans. Automatic Control, Vol. AC-29, pp 376-379, 1984

MATRIX PRE-CONDITIONING ON OPTICAL LINEAR ALGEBRA PROCESSORS

ANJAN K. GHOSH
Department of Electrical and Computer Engineering
The University of Iowa
Iowa City, Iowa 52242

ABSTRACT

For the realization of conjugate gradient algorithm, matrix pre-conditioning is an important task which should be done at a high speed. The process of pre-conditioning can withstand computational errors and is robust. Therefore, matrix preconditioning is an ideal candidate for simple optical linear algebra processors which are fast and parallel, but may have physical sources of inaccuracies. In this paper, we describe a method for matrix pre-conditioning on parallel optical processors.

1. INTRODUCTION

Optical processors capable of manipulating discrete data are known as the optical linear algebra processors (OLAP's). Many implementations of OLAP's have been described in the last few years [1-3]. Most of these machines are data-flow type vector processors in which matrix multiplications are carried out optically in a parallel fashion. Programmable digital processors attached to the optical matrix multipliers monitor all other operations, such as, timing control, data input, output, and storage [4]. It has been shown that solution of simultaneous linear algebraic equations by gradient algorithms, or by direct methods like Gaussian elimination, QR decomposition, etc., can be carried out in an almost systolic fashion on the OLAP's [4-8]. Also, the eigenvalues and eigenvectors of matrices can be calculated on these parallel optical processors [9].

Because of the inherent parallelism and interconnection properties of light, the optical multiplication of two (NxN) matrices can be carried out in order of N or N^2 steps on the OLAP's [4-8]. On serial digital computers such a matrix multiplication requires $O(N^3)$ steps. Similar speed-up and efficiency can be achieved in the optical realization of other matrix computations. Thus, even the most rudimentary OLAP's can perform matrix computations at several hundred magaflops rate [3]. Therefore, many researchers are striving to realize and solve engineering problems involving intensive computations of linear algebra (for example, solution of partial differential equations using finite-element techniques, calculation of optimal controls, Kalman filtering, and pattern recognition by discriminant functions) on OLAP's [1,3].

Since the parallel optical matrix multiplication process is mostly analog, the matrix product calculated optically may be affected by noise and spatial wavefront errors [10]. This limits the computational accuracy of simple, low-cost optical processors and overshadows their potential for high speed parallel computation.

Thus, the most effective way of using these OLAP's is to utilize them in implementing robust computations which can tolerate inaccuracies and which require high speed and parallelism. Matrix pre-conditioning is one such algorithm. Pre-conditioning of matrices is essential for solution of equations or optimization problems with the powerful and elegant conjugate gradient algorithm [11]. In this paper, we describe how to design an optical preprocessor to perform the task of matrix pre-conditioning. Such an optical preprocessor will be a very important component of an OLAP for realization of the conjugate gradient algorithm.

In Section 2, the basic technique of two-dimensional optical matrix-vector multiplication is described. The techniques of matrix pre-conditioning to speed up the convergence of gradient algorithms are discussed in Section 3. A parallel optical realization of matrix pre-conditioning is discussed in Section 4. Section 5 makes the concluding remarks.

2. OPTICAL MATRIX PROCESSORS

The basic element of an OLAP is the optical matrix multiplier with an optical mask which stores the elements of a given data matrix. Such a mask can be made from photographic film or liquid crystal array or any other electro-optic, acousto-optic or magneto-optic device. The concept of parallel optical matrix-vector multiplication is described with reference to Figure 1. In this figure, the output intensities $\{a_m\}$ of N input LED's or laser diodes describe a vector a. This plane is imaged vertically and expanded horizontally (through bundles of optical fibers or lens array) such that light leaving the m-th LED (a_m) uniformly illuminates the m-th row of the two-dimensional mask. The transmittances b_{nm} of the mask describe the (NXN) matrix B^T. The light distribution leaving the mask is then summed vertically on a linear photodetector array. The output vector c^T is the matrix-vector product [4]

$c^T = a^T B^T$ or $c = Ba$.

Thus, here all the N^2 multiplications $b_{nm}a_m$, for $1 \leq n,m \leq N$, are performed in parallel by the passage of the light of LED's through the mask. The addition of numbers is performed by the focussing of light leaving a column of the mask onto a photodetector. With parallel addressing of the LED's and photodectors, we require only one clock cycle to perform the entire (NXN) matrix -(N) vector multiplication. Neglecting the set-up time, the speed-up from optical parallelism is $O(N^2)$.

Once the vector c is calculated optically it can be processed or fed back to LED's by suitable feedback paths. Controlling the post-processing and the data flow, we can realize the various algorithms for matrix decompositions, solution of linear algebraic equations, or eigenvalue computations mentioned before [1,4-8]. In these optical realizations of various algorithms, speed-up factors vary from O(N) to $O(N^2)$ depending on the architecture and the data flow schemes.

Three main components of the optical matrix multiplier described in Fig.1 are (i) the input array of light sources, (ii) the spatial light modulator, and (iii) the output photodetector array. Each of these optical components contains various spatial errors (nonuniformities, attenuations, nonlinearities, dark current, etc.). These spatial wavefront errors affect the accuracy of the matrix multiplication. The random noise in the photodetectors also affects the results. These errors and their effects on computational accuracy of the various algorithms have been analyzed and quantized in Refs. [10,12]. Low-cost experimental optical processors have an accuracy of three or four significant digits. With very careful design the accuracy can be increased to approximately seven or eight significant digits [3]. It is impossible to eliminate the errors altogether in these "partly analog" optical processors even though they may offer speed-up of $O(N^2)$. Thus to utilize these OLAP's effectively, we should either (1) desensitize the data and the problem with respect to the processor errors or (2) implement only robust algorithms like pre-conditioned conjugate gradient method on the OLAP's.

3. MATRIX PRE-CONDITIONING

The condition number of a matrix A is defined as $C(A) = \|A\| \, \|A^{-1}\|$ [11]. Computational inaccuracies in linear algebra problems involving matrix inversion or solution of linear equations are proportional to the condition number C(A) and the errors in the processor hardware. The rate of convergence of iterative algorithms depends on the condition number: the smaller the condition number, the faster the convergence.

Therefore, when the conjugate gradient algorithm is used to solve a linear system of equations,

$$Ax = b, \tag{1}$$

(where A is the NxN coefficient matrix, b is a known N-vector, and x is the unknown) we pre-condition the problem to reduce the effective condition number [11]. Pre-conditioning of the problem involves the computation of a matrix M^{-1}, and multiplication of both sides of (1) by M^{-1} to obtain a modified system of equations:

$$M^{-1}Ax = M^{-1}b. \tag{2}$$

The nonsingular matrix M is basically an approximation of A such that $M^{-1}A$ has a "small" condition number $C(M^{-1}A) < C(A)$ or has small rank. The inversion of M should be relatively easy.

Several possible ways to obtain a suitable M have been proposed [11,13,14]. The three most popular methods are:

(1) Take M to be the diagonal, tridiagonal, or small bandwidth part of A.
(2) Obtain M from an "incomplete" Cholesky decomposition of A such that

the sparsity pattern of A and $M^{-1}A$ are the same.

(3) Calculate M as a splitting matrix for some suitable linear iterative method like Jacobi or SOR algorithm [13].

We found that the technique (3) is the most suitable method for realization on parallel optical processors. Without loss of generality we assume that the linear iterative method used in the calculation of M is Jacobi algorithm. Then the splitting of A which defines Jacobi iterations is

$$A = D-Q,$$

where D is a diagonal matrix whose elements are the same as those on the diagonal of A. Jacobi iterations for $i \geq 0$ are described by

$$Z_{i+1} = D^{-1}(QZ_i+b). \tag{3}$$

Z_0 is any arbitrary initial choice for solution of the linear system of equations $Ax = b$ in (1). It can be shown easily that for $Z_0 = 0$, i-th iteration of (3) essentially calculates

$$Z_i = (I+G+G^2 + \ldots + G^{i-1})D^{-1}b, \tag{4}$$

where I is the identity matrix and $G = D^{-1}Q$. If A satisfies the convergence requirements for Jacobi iterations [11], then as $i \to \infty$,

$$Z_i = (I+G+G^2+\ldots)D^{-1}b \to A^{-1}b. \tag{5}$$

It is easy to see that if we continue the iterations (p+1) times, we generate a matrix

$$M^{-1} = (I+G+G^2 +\ldots+G^p)D^{-1} \tag{6}$$

as an approximation of the desired A^{-1}. M^{-1} defined in (6) is a valid pre-conditioning matrix because $C(M^{-1}A) < C(A)$. Of course, in the limit $r \to \infty$, $M^{-1} \to A^{-1}$ and $C(M^{-1}A) \to C(I) = 1$. Hence, the smaller the condition number $C(MA^{-1})$ we desire, the more computations we have to perform by keeping p large. There is clearly a trade-off.

4. REALIZATION OF PRE-CONDITIONING ON OPTICAL PROCESSORS

There are two ways in which the pre-conditioning operation can be incorporated in the iterations of the conjugate gradient algorithm:

(A) Calculate M^{-1} as shown in (6) and then multiply both sides of $Ax = b$ with M^{-1} to obtain the pre-conditioned system of equations as in (2). Then start the loop of conjugate gradient iterations with the pre-conditioned system of equations.

(B) In the j-th ($j \geq 0$) iteration of conjugate gradient calculate a modified residual error $\hat{r}_j$ by solving the set of equations

$$A\hat{r}_j = r_j, \tag{7}$$

where $r_j = (b - Ax_j)$ is the residual error in the j-th iteration of conjugate gradient algorithm and x_j is the j-th iterate of the conjugate gradient algorithm [13]. We want $x_j \rightarrow x = A^{-1}b$ as j increases. It is easy to see that if (7) is solved by (p+1) iterations of the Jacobi iterations, starting with an initial choice $\hat{r}_j(0) = 0$, then

$$\hat{r}_j(p+1) = (I+G+G^2+\ldots+G^p)D^{-1}r_j = M^{-1}r_j = M^{-1}b - M^{-1}Ax_j. \tag{8}$$

Thus $\hat{r}_j(p+1)$ is the residual of the properly pre-conditioned problem and can be used to update the (j+1)-th iterate of the conjugate gradient algorithm.

Even though the technique (B) seems involved, it is the preferred method for executing pre-conditioned conjugate gradient. With this technique, we need to perform extra p matrix-vector multiplications during each iteration of the conjugate gradient loop. The p matrix-vector multiplications arise in an inner loop for calculating $\hat{r}_j(p+1)$ with Jacobi iterations. This inner loop for Jacobi iteration can be very easily implemented on an OLAP with parallel feedback paths from the photo-detector and post-processor to the LED's [4,12].

Let us look at equation (3) to understand the optical realization of Jacobi iterations. The iteration matrix $G = D^{-1}Q$ is calculated first in the digital part of the OLAP and then it is stored in the optical mask shown in Fig. 1. The vector $D^{-1}b$ can also be calculated in the digital part and is stored in the RAM of that processor. Then Jacobi iterations are started. The iterate Z_i is input to the LED's in parallel and in the same clock cycle the optical matrix-vector product GZ_i is available at the detector plane. The product GZ_i is then passed to the digital postprocessor. The postprocessor adds the stored vector $D^{-1}b$ to GZ_i in parallel to form the new iterate $Z_{i+1} = (GZ_i+D^{-1}b)$ as shown in (3). The new iterate Z_{i+1} is then fed back to the LED's in the front-end of the optical multiplier (Fig. 1) and the next iteration is started. Irrespective of the dimension N of the matrix and vectors, the parallel optical matrix-vector multiplication requires O(1) time-steps. Thus, from (8) we find that the pre-conditioned residual $\hat{r}_j(p+1)$ can be calculated in parallel in O(p) time-steps on an OLAP. On a serial digital computer the same computations require $O(pN^2)$ steps, and thus, an optical realization of pre-conditioning can be $O(N^2)$ faster.

The method (A) for pre-conditioning (from (6)) requires computation of (p-1) matrix-matrix products G^2, G^3, ..., G^p and $M^{-1}A$ (without considering the trivial multiplications with diagonal D^{-1} matrix). The matrix G can be stored on the optical mask of Fig. 1. Then, if each column of G is fed successively to the LED's, the columns of the product G^2 would appear on the detector array. Thus, the optical calculation of G^2 would require O(N) time-steps. The columns of G^2 can be again fed back in parallel to the LED's to perform the computation of G^3. After (p-2) sets of such optical matrix-matrix multiplications, we obtain the terms G^2, G^3, ..., G^p in O(pN-2N) steps. The optical computation of M^{-1} and $M^{-1}A$ and $M^{-1}b$ would then require O(N) steps. Again, we notice that calculation of all these matrix-matrix multiplications would require $O(pN^3)$ steps on serial digital computers.

5. DISCUSSION

Pre-conditioning is a crucial and time consuming step in the implementation of conjugate gradient algorithm [11,15]. Thus, fast and parallel optical realization of pre-conditioning will be helpful. The rest of the computations of the conjugate gradient algorithm can then be performed on a simple digital processor. It is also possible to perform those computations on an OLAP.

As we have mentioned in Sections 1 and 2, the noise and spatial errors of the optical matrix-vector multiplier would affect the calculation of pre-conditioning matrix M^{-1}. We obtain a pre-conditioning matrix $(M+E)^{-1}$ optically, where E is an error matrix. This $(M+E)^{-1}$ would be less effective in pre-conditioning the original data matrix A, but E would not affect the final solution $x = A^{-1}b$ obtained by the conjugate gradient algorithm. For example, assume that C(A) of the original A is 10^4 and the conjugate gradient algorithm takes 300 iterations to converge. If we calculate an M^{-1} such $C(M^{-1}A) = 20$, then the conjugate gradient algorithm would converge to the same accuracy in 15 iterations. However, on optical processor, the same M^{-1} would be calculated with some error, and, so, with optical pre-conditioning we obtain $C(M^{-1}A) = 35$. Then, the conjugate gradient algorithm would converge to the same accuracy in 20 iterations. Thus the price to pay for the small errors in optical pre-conditioner would be a few extra steps in conjugate-gradient algorithm but not the accuracy of the final solution.

Pre-conditioning of matrix data may be very helpful with other algorithms of linear algebra in reducing the computational load and improving the accuracy of the solution. The optical pre-conditioners we described can be used in these cases as pre-processors to desensitize the data to round-off and other errors. These optical pre-conditioners can be readily incorporated in the bimodal optical computers [16] to improve their efficacies even further. In fact it is easy to envision "trimodal" optical computers with (i) an optical pre-conditioner, (ii) a high speed optical linear algebra processor, and (iii) a digital postprocessor. Such trimodal processors will incorporate the best features of analog optics, namely, advantages in speed, size, power consumption and parallelism, and of digital electronics, namely flexibility. Many studies remain to be performed on pre-conditioning strategies, algorithms and hardware.

REFERENCES

[1] Special issue on Optical Computing, Proc. IEEE, Vol. 72, July 1984.

[2] Special issue on Optical Computing, Optical Eng., Vol. 24, Jan. 1985.

[3] Special issue on analog Optical Computing, Appl. Optics, Vol. 26, July 1986.

[4] M. Carlotto and D. Casasent, Appl. Optics, Vol. 21, pp. 348-350, Nov. 1982.

[5] D. Casasent and A. Ghosh, Proc. SPIE, Vol. 388, pp. 182-189, Jan. 1983.

[6] A. Ghosh and D. Casasent, Appl. Optics, Vol. 22, pp. 1795-1796, June 1983.

[7] D. Casasent and A. Ghosh, Optics Comm., Vol. 46, pp. 270-273, July 1983.

[8] D. Casasent and A. Ghosh, Appl. Optics, Vol. 22, pp. 3572-3578, Nov. 1983.

[9] H. Caulfield, et al, Appl. Optics, Vol. 20, pp. 2263-2265, July 1981.

[10] D. Casasent and A. Ghosh, Optics Letters, Vol. 10, pp. 252-254, June 1985.

[11] G. Golub and C. VanLoan, Matrix Computations, Johns Hopkins Press, Baltimore, MD, 1983.

[12] A. Ghosh, Ph.D. dissertation, Carnegie-Mellon University, Pittsburgh, PA, 1984.

[13] J. Ortega and R. Voigt, Siam Review, Vol. 27, pp. 149-240, June 1985.

[14] D. Evans, E. Lipitakis and N. Missirlis, Int. Journal Computer Maths.Sec. B, Vol. 9, pp. 55-80, 1981.

[15] A. Sameh, in Computer Phys. Comm., Vol. 37, pp. 159-166, 1985.

[16] H. Caulfield, et al., Appl. Optics, Vol. 25, pp. 3128-3131, Sept. 1986.

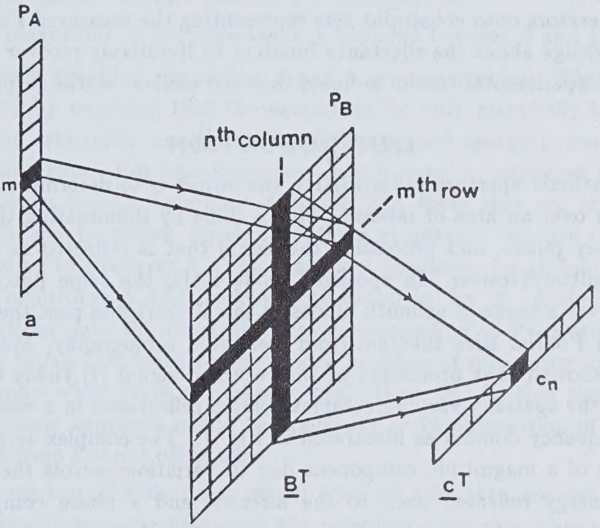

P_A
P_B
n^{th} column
a_m
m^{th} row
P_C
$\underline{a}$
c_n
$\underline{B}^T$
$\underline{c}^T$

Figure 1

Parallel Optical Matrix-vector Multiplication (c=Ba)

RECONSTRUCTION OF FREQUENCY-OFFSET FOURIER DATA BY ALTERNATING PROJECTION ONTO CONSTRAINT SETS *

C. A. WEBER
Gallagher Associates, Inc., 2639 South River Road, West Lafayette, IN 47906
J. P. ALLEBACH
School of Electrical Engineering, Purdue University, West Lafayette, IN 47907

ABSTRACT

Spotlight mode synthetic aperture radar (SAR) provides measurements in a limited annular sector of the 2D spatial Fourier transform of the complex reflectance function of a spatially limited terrain patch. Modelling the complex terrain reflectance function as consisting of a smooth narrowband, lowpass magnitude component multiplied by a rough broadband phase component, we employ a sucession of projection operators onto constraint sets representing the measured Fourier data and *a priori* knowledge about the rflectance function to iteratively recover the reflectance magnitude. Experimental results indicate that the method works surprisingly well.

1. INTRODUCTION

With synthetic aperture radar (SAR), the object is to determine the reflectance of the terrain over an area of interest. This is done by illuminating the terrain with radio frequency pulses, and processing the signal that is reflected or scattered back to the transmitter/receiver. In spotlight mode SAR, the same patch of terrain is illuminated over a range of azimuth angles as the aircraft flies past the terrain patch. Based on the Fourier slice theorem from computed tomography, Munson, O'Brien, and Jenkins showed that processing of the returned signal [1] yields the 2D Fourier transform of the spatially varying complex terrain reflectance in a small annular sector of the frequency domain as illustrated in Fig. 1. The complex terrain reflectance itself consists of a magnitude component due to variations across the terrain in the amount of energy reflected back to the aircraft, and a phase component due to interaction at the air/surface interface and variations in surface height across the terrain [1],[2].

We model the magnitude component as varying smoothly, the phase component as fluctuating rapidly, and consider the problem of recovering the magnitude component from the limited region of Fourier data shown in Fig. 1. The complex terrain reflectance function results from multiplication of these two components in the spatial domain. In the frequency domain, the spectrum of the narrowband, low-pass magnitude component is convolved with that of the wideband phase component, thereby spreading the spectral energy in the magnitude component over a very broad band of frequencies. The problem is then to recover the magnitude component from a limited portion of this broad spectral band.

---

* This work was partially supported by the National Science Foundation under Grant ECS-8419997.

In our approach to this problem, all known information about the signal to be recovered is represented in the form of constraint sets. For each constraint set, we define a constraint operator that maps the signal into the corresponding constraint set. If the signal is already a member of the set, it is unchanged by the operator. We then apply each of these operators in succession to the signal, repeating the process until satisfactory convergence is obtained. Algorithms of this type have been successfully used in many signal design and recovery problems. If the constraint sets are convex and the operators project the signal to the closest member of the constraint set, it is known that the algorithm will converge [3]. For our problem, the constraint sets represent the following information about the complex reflectance:

1. Its Fourier transform matches the known data over the measurement region.
2. The spatial support of the reflectance is limited to the area of the patch illuminated by the transmitter.
3. The magnitude of the reflectance is bandlimited.
4. The magnitude of the reflectance is bounded between 0 and 1.

Note that, strictly speaking, Properties 2 and 3 are inconsistent. This problem can be circumvented by requiring that the magnitude be only *essentially bandlimited* [4] *i.e.* all but some arbitrarily small amount of the spectral energy is contained within the chosen bandregion. For simplicity, we do not concern ourselves here with the distinction between strictly bandlimited signals and those that are only essentially bandlimited. It can be shown [3] that all these constraint sets are convex except that corresponding to Property 3. The constraint operators for all four sets are projections. The algorithm is described in more detail in Sec. 2. A similar algorithm utilizing projections onto the constraint sets corresponding to Properties 1, 2, and 4 was applied by Sezan and Stark to the related problem of recovering images in computed tomography when there are missing projections [5]. In that paper, they also suggested that their approach would be applicable to reconstruction of complex terrain reflectance from Fourier offset data.

Although our terrain reflectance model does not take into account the vectorial nature of the electromagnetic fields and the complete scattering matrix that characterizes reflection of these fields from the terrain surface, this does not preclude the alternating projections onto constraint sets algorithm from significantly improving the quality of SAR imagery. In addition, this model and our algorithm are also applicable to speckle noise removal in other types of coherent imaging systems, including those operating at wavelengths in the visible.

2. ALTERNATING PROJECTION ONTO CONSTRAINT SETS

For simplicity, we describe the algorithm in terms of 1D rather than 2D signals. For the same reason we use a continuous-parameter setting. Extension to the 2D and discrete-parameter cases is straightforward. The terrain reflectance function is denoted by

$$g(x) = m(x)\,p(x)\,, \tag{1}$$

where m is the narrowband magnitude function that is to be recovered, and

$$p(x) = \exp[i\phi(x)] \ , \tag{2}$$

is the wideband component due to the phase function ϕ. Letting uppercase letters denote the Fourier transforms

$$G(u) \triangleq \mathcal{F} g(x) = \int_{-\infty}^{\infty} g(x) \exp[-i2\pi ux] dx \ , \tag{3}$$

of signals denoted by the corresponding lower case letters, we have that

$$G(u) = M(u) * P(u) \ . \tag{4}$$

Each known property of the signal g is expressed in terms of a constraint set C such that $g \in C$ if and only if g has the prescribed property. The constraint sets for our problem are:

$$C_1 = \{g(x): \ G(u) = G_0(u), \ u \in \Omega\} \ , \tag{5}$$

where Ω denotes the set of frequencies at which measurements have been obtained, and G_0 denotes the measured value of the reflectance transform at those frequencies;

$$C_2 = \{g(x): \ g(x) = 0, \ x \notin S\} \ , \tag{6}$$

where S is the support region of the terrain reflectance function;

$$C_3 = \{g(x): \ M(u) = 0, \ u \notin \Gamma\} \ , \tag{7}$$

where Γ is the support region of the magnitude reflectance transform; and

$$C_4 = \{g(x): \ 0 \le m(x) \le 1\} \ . \tag{8}$$

For each constraint set C, we define a constraint operator P which for any signal f, satisfies

$$Pf \in C, \text{ if } f \notin C \ , \tag{9a}$$

$$Pf = f, \text{ if } f \in C \ . \tag{9b}$$

The constraint operators corresponding to the constraint sets C_j, $j = 1, \ldots, 4$ are:

$$P_1 f = \mathcal{F}^{-1} \tilde{P}_1 \mathcal{F} f \ , \tag{10a}$$

where

$$\tilde{P}_1 F(u) = \begin{cases} G_0(u), & u \in \Omega, \\ F(u), & \text{else;} \end{cases} \tag{10b}$$

$$P_2 f(x) = \begin{cases} f(x), & x \in S, \\ 0, & \text{else;} \end{cases} \tag{11}$$

$$P_3 f = e^{i \arg f} \{\mathcal{F}^{-1} \tilde{P}_3 \mathcal{F} |f|\} \ , \tag{12a}$$

where

$$\tilde{P}_3 H(u) = \begin{cases} H(u), & u \in \Gamma, \\ 0, & \text{else;} \end{cases} \tag{12b}$$

and

$$P_4 f = e^{i \arg f} \{\tilde{P}_4 |f|\} , \tag{13a}$$

where

$$\tilde{P}_4 h(x) = \begin{cases} 0, & h(x) < 0, \\ h(x), & 0 \leq h(x) \leq 1, \\ 1, & 1 < h(x). \end{cases} \tag{13b}$$

Our definition of a constraint operator P for the constraint set C only requires that a signal $f \notin C$ be mapped into any signal in C. However, of all signals in C, it is natural to choose that signal which is closest to f in some sense. As a measure of the distance between two signals f and h, we choose $||f-h||$, where the norm

$$||h|| = \langle h,h \rangle^{1/2} , \tag{14}$$

is defined in terms of the inner product

$$\langle f, h \rangle = \int f(x) h^*(x) dx . \tag{15}$$

We can then say that a constraint operator P projects f to the closest signal in C if

$$||Pf - f|| \leq ||h - f|| , \text{ for all } h \in C . \tag{16}$$

It can be shown that the constraint operators P_j, $j = 1, \ldots, 4$ are, in fact, all projection operators.

In terms of these operators, the recovery algorithm is given by

$$g_{k+1} = P_4 P_3 P_2 P_1 g_k , \tag{17}$$

where g_k is the result of the k-th iteration of the algorithm. Initially, we choose

$$g_0 = F^{-1} G_0 , \tag{18}$$

the inverse Fourier transform of the measured data. The behavior of iterative schemes of the form of (17) has been of considerable interest to mathematicians [6]; and there are a number of approaches to investigating their convergence properties. Of particular interest here are the results of Youla and Webb [3] who proved convergence under the conditions that the operators in (17) be projections and the constraint sets be convex. A set C is said to be convex if for any two signals $f, h \in C$, and every value of the scalar $0 \leq \lambda \leq 1$,

$$\lambda f + (1 - \lambda) g \in C . \tag{19}$$

It can be shown that the constraint sets C_1, C_2, and C_4 are convex, whereas the set C_3 is not convex. Thus, our recovery algorithm does not quite meet Youla and Webb's conditions for convergence. Nonetheless, in our experimental work with the algorithm thus far, we have always observed convergence. Some of these results are described in the next section.

3. EXPERIMENTAL RESULTS

We implemented a discrete-parameter 1D version of the projections onto constraint sets (POCS) algorithm described in Sec. 2 using the discrete Fourier transform (DFT). We chose as our terrain reflectance magnitude signal m_n that shown in Fig. 2(a). This signal has been previously studied in connection with spectral estimation and extrapolation [7],[8]. The support interval consists of 64 of the total 256 points in the signal domain. The DFT of m_n is shown in Fig. 2(b). It is approximately bandlimited to the interval $\Gamma = [-10,10]$ which contains over 99.5% of its total energy. We generated a sequence of 256 independent random variables uniformly distributed on $[-\pi, \pi]$ for the phase ϕ_n of the complex terrain reflectance g_n. Fig. 2(c) shows a typical spectrum for the complex terrain reflectance obtained by computing G_k , the DFT of $g_n = m_n e^{i\phi_n}$. In Figs. 3, 4, 5, and 6, we investigate recovery of m_n from measurements of G_k in a band centered at $k = 60$. The width Ω of this band varied from $0.25\underline{\Gamma}$ to $4.0\underline{\Gamma}$, where $\underline{S}$ denotes the number of points in the set S. Figs. 3(a), (b) and (c) show the signals obtained by directly inverting three measurement bands of Fourier data spanning this range. In Figs. 3(d), (e) and (f), we have the signals obtained from the corresponding measurement bands after 200 iterations of POCS. The first measurement band corresponds to $\underline{\Omega}/\underline{\Gamma} = 0.25$. Direct inversion of this limited portion of Fourier data results in the signal shown in Fig. 3(a) which does not even resemble the basic form of m_n in Fig. 2(a). By comparision, the signal in Fig. 3(d), which is the result of POCS, contains 2 distinct peaks similar to m_n. Expanding the measurement band to $\underline{\Omega}/\underline{\Gamma} = 1.0$ causes an improvement in the shape of the signal recovered by direct inversion. However, many spurious peaks are present, as seen in Fig. 3(b). The signal shown in Fig. 3(e), recovered from the same measurement band but by POCS, shows that the spurious peaks have been removed, and the signal more closely resembles the original. These results compare very favorably with those recently reported by Munson and Ullman using a noniterative minimum norm least squares extrapolation algorithm [8]. It is of particular interest that in contrast to their algorithm, the quality of the signal recovered by our algorithm continues to improve as $\underline{\Omega}/\underline{\Gamma}$ exceeds unity. This is illustrated by Figs. 3(c) and (f) which show the signals recovered from spectral measurements over a band for which $\underline{\Omega}/\underline{\Gamma} = 4.0$. Considering the 3 signals shown in Figs. 3(d), (e) and (f), which were recovered by POCS, we observe that they all have the same basic shape as that of m_n. But as the width of the measurement band increases, the energy in the recovered signal increases.

To strengthen these observations, we calculate the Signal-to-Error Ratio

$$\mathrm{SER} = 10\log_{10}\left[\frac{||m||^2}{||m-\hat{m}||^2}\right]\mathrm{dB}, \tag{20}$$

as a quantitative measure of the quality of the recovered terrain reflectance magnitude $\hat{m}_n$. Because the results of both direct Fourier inversion and POCS vary significantly depending on the particular realization of the phase sequence, we calculate the average SER based on reconstructions from 10 different phase sequences.

The SER in (20) is a measure of gross similarity between m and $\hat{m}$. As a specific measure of similarity in shape between m and $\hat{m}$, we scale $\hat{m}$ by the factor

$$\eta = \frac{<m, \hat{m}>}{||m||^2}, \tag{21}$$

which minimizes $||m - \eta\hat{m}||$, and calculate the SER for $\eta\hat{m}$. A plot of terrain reflectance magnitude SER for scaled and unscaled signals from direct inversion and POCS is shown in Fig. 4. With POCS, we observe a 6-8 dB improvement in SER for the scaled signal over direct inversion for all values of Ω/Γ. This is consistent with the appearance of the reconstructions in Fig. 3. For the unscaled signals, we observe a steady increase in SER with increasing Ω/Γ. These results demonstrate that POCS effectively recovers the terrain magnitude reflectance function from offset Fourier data. However, POCS does not recover the complex terrain reflectance g_n. Fig. 5 shows that SER's for scaled and unscaled $\hat{g}_n$'s recovered by both direct inversion and POCS are all extremely low for all values of Ω/Γ. Nor does POCS recover the total energy in the terrain reflectance as indicated by Fig. 6.

4. CONCLUSIONS

If one views our recovery algorithm as a scheme for extrapolating the spectrum G of the complex reflectance function, then the experimental results presented above seem unreasonably good. Properties 2 - 3 in Sec. 1 certainly do not represent sufficient *a priori* information to justify the expectation of recovering the broadband complex reflectance function g from only a limited portion G_0 of its spectrum. On the other hand, if one views the algorithm as a means of recovering only the narrowband reflectance magnitude m, then our results seem more plausible. After all, convolution of the magnitude spectrum M with the spectrum P of the phase component simply smears M over a broad spectral band. In this case, one might argue that any portion of the spectrum G with extent $\geq$ the support of M contains all the information in M, albeit in an unknown encoded form. Indeed, the problem is akin to demodulation of a direct sequence spread spectrum signal when the modulating sequence is unknown. As the length of the measurement band increases beyond the support of M, the information about M appears redundantly, making the decoding process easier. This is consistent with our observed results. Another analogy can be drawn with holography using diffuse illumination or rough, scattering objects. In this case, it is well known that the object can be reconstructed from only a portion of the hologram recording, but at lower spatial resolution, and with a more restricted viewing angle [7]. In fact, Munson and Sanz [2] drew upon this analogy as a way of explaining how SAR images could be reconstructed from Fourier offset data.

REFERENCES

[1] D. C. Munson, J. D. O'Brien, and W. K. Jenkins, "A Tomographic Formulation of Spotlight Mode Synthetic Aperture Radar," *Proc. IEEE,* vol. 71, pp. 917-925, Aug. 1983.

[2] D. C. Munson and J. L. C. Sanz, "Image Reconstruction from Frequency - Offset Fourier Data," *Proc. IEEE,* vol. 72, pp. 661-669, June 1984.

[3] D. C. Youla and H. Webb, "Image Restoration by the Method of Convex Projections - Part 1," *IEEE Trans. Med. Imaging,* vol. MI-1, pp. 81-94, Oct. 1982.

[4] D. Slepian, "On Bandwidth," *Proc. IEEE,* vol. 64, 292-300, 1976.

[5] M. I. Sezan and H. Stark, "Tomographic Image Reconstruction From Incomplete Information By Convex Projections and Direct Fourier Inversion: Suggested Applications to Spotlight-mode Synthetic Aperture Radar (SAR) Imaging," *Proc. of the 18th Annual Conference on Information Sciences and Systems,* Princeton, NJ, March 14-16, 1984, pp. 72-76.

[6] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables,* New York: Academic Press, 1970.

[7] A. K. Jain and S. Ranganath, "Extrapolation Algorithms for Discrete Signals with Application in Spectral Estimation," *IEEE Trans. Acoustics, Speech, Signal Processing,* vol. ASSP-29, pp. 830-845, August 1981.

[8] D. C. Munson and E. A. Ullman, "Support-Limited Extrapolation of Offset Fourier Data," *Proceedings of the 1986 International Conference on Acoustics, Speech, and Signal Processing,* Tokyo, Japan, 7-11 April 1986, pp. 2483-2486.

[9] R. J. Collier, C. B. Burckhardt, and L. H. Lin, *Optical Holography,* New York: Academic Press, 1971.

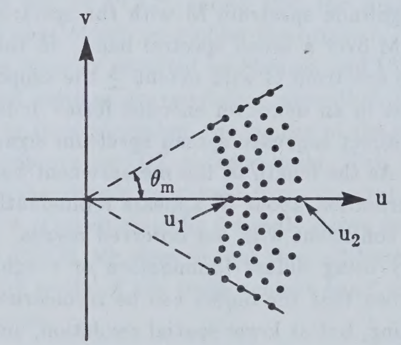


Fig. 1. Region of offset Fourier data measured with spotlight mode synthetic aperture radar.

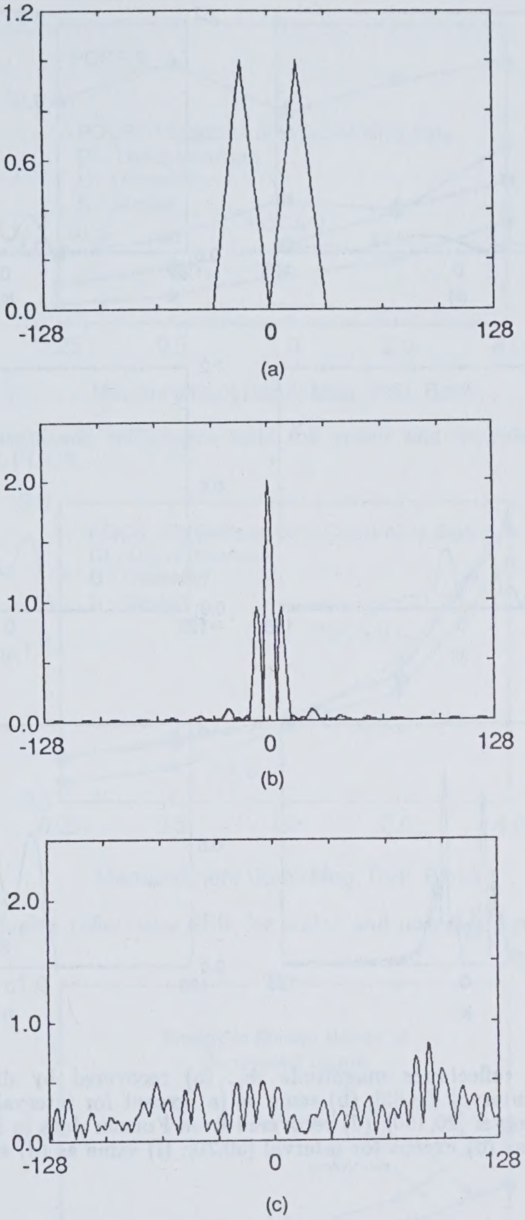


Fig. 2. Test signal: (a) terrain reflectance magnitude m_n; (b) its spectrum M_k; (c) spectrum G_k of complex reflectance function g_n.

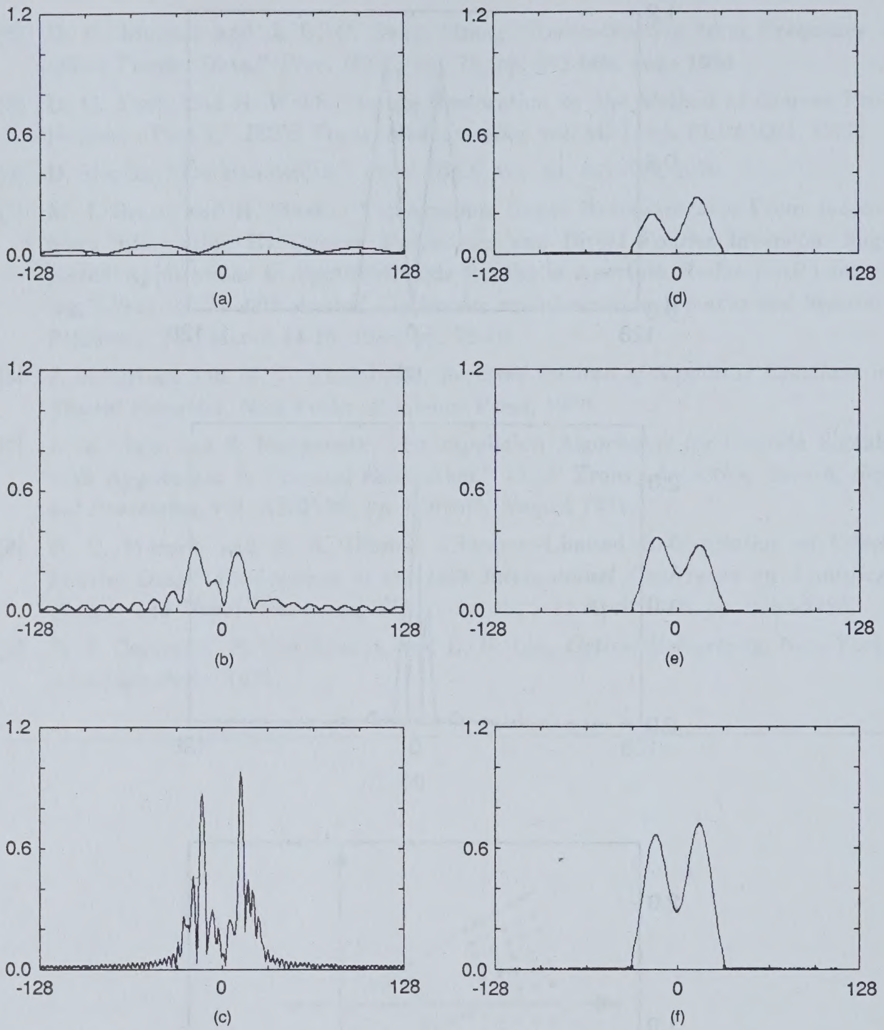


Fig. 3. Terrain reflectance magnitude $\hat{m}_n$: (a) recovered by direct inversion of Fourier data in interval [58,62]; (b) same as (a) except for interval [50,70]; (c) same (a) except for interval [20,100]; (d) recovered from Fourier data in interval [58,62] by POCS; (e) same as (d) except for interval [50,70]; (f) same as (d) except for interval [20,100].

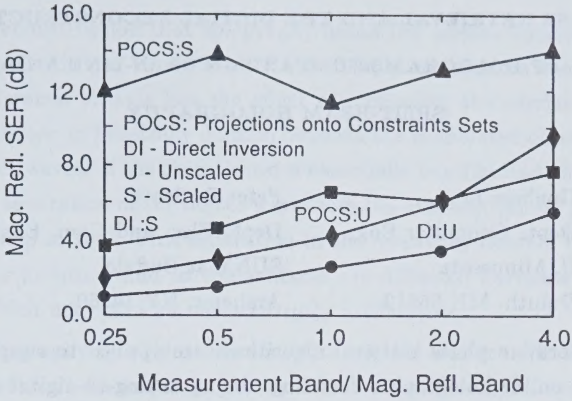


Fig. 4. Average magnitude reflectance SER for scaled and unscaled signals after direct inversion and POCS.

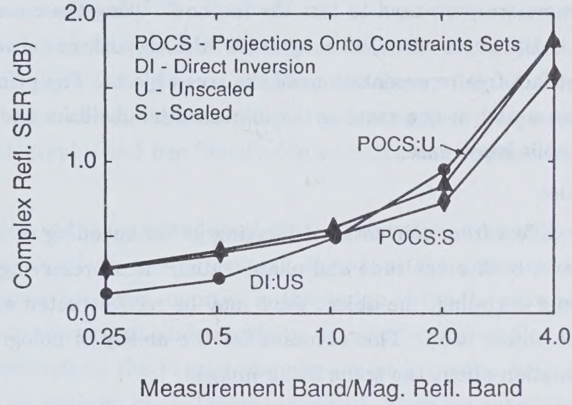


Fig. 5. Average complex reflectance SER for scaled and unscaled signals after direct inversion and POCS.

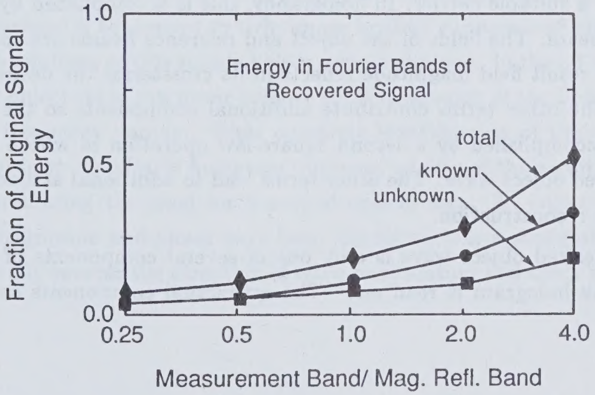


Fig. 6. Energy in Fourier bands of recovered signal.

PHASE RETRIEVAL AND THE DIGITAL RECONSTRUCTION OF HOLOGRAMS:COMPARISON OF IN-LINE AND SPLIT-BEAM HOLOGRAPHY

Gonhsin Liu
Dept. Computer Eng.
U. Minnesota
Duluth, MN 55812

Peter D. Scott
Dept. Elec. and Com. Eng.
SUNY at Buffalo
Amherst, NY 14260

ABSTRACT—Iterative phase retrieval algorithms are applied to suppress artifact in split beam and online holographic decoding. Using analog-to-digital converted holograms, the algorithms iterate between an object domain in which the signal is constrained to match the digitized hologram. Simulated holograms with known objects and optical parameters were used to test the method. Using the conventional reconstruction as an initial guess on object magnitude, the algorithms converged monotonically toward artifact-free representations of the true objects. The principal difference between split-beam and in-line cases is the increased bandwidths and film resolution required in the split beam case.

1. Introduction

Holography differs from conventional imaging in the encoding of a complete copy of the object wave, both magnitude and phase, rather than preserving the magnitude only. With proper decoding, the object wave may be reconstructed exactly as it was scattered off the object itself. This accounts for the ability of holograms to capture full of 3D information about the scene being imaged.

Recording of both magnitude and phase by media(photographic plates, video cameras,etc.) sensitive to magnitude only can only be accomplished by modulation of the object wave by a suitable carrier. In holography, this is accomplished by introduction of a reference beam. The fields of the object and reference beams are spatially superposed, and the result field magnitude reflects in its crossterms the desired square-law modulation. The other terms contribute additional components to the hologram, is itself usually accomplished by a second square-law operation in whose crossterms is found the desired object wave. The other terms lead to additional artifact which limit visibility of the reconstruction.

Thus the desired object wave is only one of several components of the net field produced as the hologram is read out. The artifactual components so degrade the quality of the reconstruction that holography found few serious applications until the development by Leith and Upatnieks[1] of a technique to suppress these components. Moving the reference off-axis has the effect of increasing the carrier frequency and creating a separation in frequency domain between the modulated object wave and the principal artifact waves. If the object wave is essentially bandlimited and the frequency shift adequate, separation of the object wave from the artifacts can be preserved in the demodulation step as well. This separation in the frequency domain implies separate beam directions in space, and so the artifacts are deflected harmless away from the object wave, which may then be viewed largely artifact-free.

Off-axis holography quickly superceded the original Gabor or in-line holography[2] in most applications. An important exception is when the object consists of an ensemble of small distinct entities(such as an aerosol or other particle field) and the hologram is taken in the far field of the individual entities. In this case the main artifact components are spatially dispersed over the reconstruction plane while the object is highly localized, and thus artifact does not obscure the actual particle boundaries[3]. Since in-line technique is in general simpler, less demanding of coherence and film resolution, and more robust than the off-axis technique, it is the preferred method for particle field holography and has found wide scientific and engineering application in this context[4].

The extraction of the magnitude and phase information from magnitude-only data is, in the digital signal processing context, known as the phase retrieval algorithm. Substantial progress has been reported in the design of the iterative procedures operating cyclically in the primal and transform domains which employ split constraints (eg. known magnitude in the Fourier domain, object known to be real and having a bounded region of support in the primal domain). Here we substitute this iterative computational technique[5] for conventional optical reconstruction as a method for reconstruction of the object wave in both in-line and off-axis holography. In the in-line case, phase retrieval is equivalent to twin image artifact suppression[6] and thus extends the practical usefulness of this simple holographic technique. In the off-axis case, modulation of the object wave can never achieve total separation of the object and artifact bands in the frequency domain. Thus complete identification of the object wave by phase retrieval from its off-axis hologram improves quality of the resulting reconstruction while eliminating the need for a second optical step. In either case, once the object wave magnitude and phase have been identified, it is a straightforward matter to algorithmically reverse the direction of wave propagation and track the object wave back to its source, the object itself.

2. Computational Model for Holography

Let the object illuminated by coherent radiation of wavelength λ produce a harmonic scalar field $a(x, y)$ in the $z = 0$ plane propagating in the $+z$ direction($a(x, y)$ and all other fields are complex functions of their spatial coordinates; the temporal factor $exp[j\omega t]$ has been suppressed throughout). Then the object field $O(x, y, \varsigma)$ scattered onto the hologram plane $z = \varsigma$ by the object luminance can be expressed as a 2D convolution

$$O(x, y, \varsigma) = a(x, y) * *h_\varsigma(x, y) \tag{1}$$

where the point spread function $h_\varsigma(x, y)$ is the quadratic phase factor

$$h_\varsigma(x, y) = \frac{1}{j\lambda\varsigma} exp[j\pi(x^2 + y^2)/\lambda\varsigma] \tag{2}$$

In addition, assume a coherent reference beam $r(x, y, z)$ of the same wavelength λ also illuminates the hologram plane $z = \varsigma$. Without loss of generality assume the direction vector for $r(x, y, z)$ lies in the $y - z$ plane so the reference beam may be expressed as

$$r(x, y, z) = r_o exp[-j\kappa((sin\theta)y - (cos\theta)z)] \tag{3}$$

In (3) κ is the wavenumber $2\pi/\lambda$. Figure 1 depicts these assumptions.

In the hologram plane $z = \varsigma$ the field is the sum $r(x, y, \varsigma) + O(x, y, \varsigma)$. The intensity of this field, or magnitude squared, is

$$I_\varsigma = r_o^2 + |O_\varsigma|^2 + r_o O_\varsigma^* exp[-j(\kappa_l y + \varphi)] + [r_o O_\varsigma^* exp[-j(\kappa_l y + \varphi)]]^* \tag{4}$$

where $O_\varsigma \equiv O(x, y, \varsigma)$, the effective wavenumber of the reference beam on the hologram plane is $\kappa_l = \kappa sin\theta$, and the phase angle $\varphi = -\kappa z cos\theta$. The first term in (4) is a constant or dc term across the entire hologram, the second gives rise to a "halo" artifact, while the last two terms show the object wave spatially modulated by a carrier with effective wavenumber κ_l.

The developed hologram will have transmission linearly related to I_ς. For simplicity take I_ς to be the hologram itself. If this hologram is placed in the object $z = 0$ and reilluminated by the plane wave $exp[j\kappa z]$, the field at $z = 0+$ is just I_ς as in (4). By linearity of the wave equation, each of the four terms may be considered a separation field propagating in the $+z$ direction. In particular consider the last term, which is the original object wave O_ς scaled by r_o and modulated by $exp[j(\kappa_l + \varphi)]$. Modulation by this complex exponential can be shown to be equivalent to deflection of the beam by the angle θ in the $y-z$ plane shown in Figure 1.[7]. Thus a reconstruction plane placed at $z=\varsigma$ will capture a virtual image of the object at an angle $-\theta$. Note that to enforce total spatial separation of these terms when the hologram size is finite is not possible due to to the angular scattering(diffraction) of each term caused by the hologram edges.

In summary, Figure 2 and 3 show the block diagrams of the in-line hologram's formation and reconstruction procedures respectively. When the tilted angle θ is selected to be zero, in-line holography is a special case of split-beam holography. The simulated object, corresponding hologram and reconstructed image are shown on the Figure 4(a), (b) and (c). A specific ς value is used for the distance between the object and hologram plane. Also, Figure 5 is the block diagram for the split-beam hologram's formation model. The reconstruction model of it could be the duplication of the formation model or just use the in-line scheme. Figure 6's (a), (b) and (c) are the split-beam holography's simulated object, hologram and its in-line reconstruction($\theta = 15^0$). For the simulation experiments shown before, the above model has been discretized and convolutions computed by fast Fourier transform. At the center of Figure 6(c) is the dc plus halo effect, to its left the twin image and to its right the real image. Note the stripping in the real image, due to the imperfect separation of these components. In the next section, a phase retrieval algorithm will be presented to improve the reconstruction.

3. Phase Retrieval Algorithms

In-line Fresnel hologram phase retrieval algorithm has been discussed[6] and its convolution-deconvolution iterative loop is shown in the Figure 7. In this holographic context, the phase retrieval algorithm is equivalent to twin image elimination. The reconstructed image is used as the estimated object to start the iteration. The constraint on the object domain could be either real or with a known magnitude. And the magnitude part of the hologram field is the constraint of the hologram field. The in-line hologram phase retrieval algorithm is then completely specified, for retrieving phase on the hologram domain, or eliminating its twin image on the reconstructed image plane. Figure 4(d) gives the 5 iteration result of the twin image elimination algorithm.

For the split-beam hologram's phase retrieval, a similar procedure can be developed. Figure 8 shows the iterative loop and initial guess of this algorithm. The initial guess is the split-beam image reconstruction with a tilted angle $-\theta$. The original formation of the holography set-up with the $+\theta$ as the angle between the reference beam and object wave. With the $-\theta$ reconstruction angle, the object will be moved back to the original position. Similar to the in-line case, the spit-beam phase retrieval algorithm is also implemented to eliminate the twin image which might affect the visibility on the reconstruction plane. This will be critical if the object's twin image is shifted and overlapped with other objects. This algorithm can also eliminate the stripes away from the objects. These stripes are due to the nonperfect separation of the dc and object terms. The forward and reverse branches of the loop are the formation and inverse procedure for the split-beam holograms. Therefore, each of the branches consists of the reference beam and the object wave. Once the initial guess is selected, the iteration will go on until the desired number of iterations are achieved. Both of the in-line and split-beam holography methods will converge. Figure 6(d), (e), (f), (g) and (h) display the algorithm results after 10, 30, 50, 75 and 100 iterations. However, there has no proof yet that both will converge to the correct solution. Usually, both algorithms converge to a data set which is approximately equal to the original object.

4. Conclusion

In this paper, we have simulated the in-line and split-beam holograms digitally. Also, we propose a set of phase retrieval algorithms for different holography. It is obvious that other holography model can also have a phase retrieval by modifying the discussed algorithms. Examples are presented to show the results of the simulations. Though both models look similar, there is still some differences between the models. Since the split-beam hologram pattern is the object wave modulated by the reference beam, the bandwidth of the split-beam hologram will be larger than the in-line hologram's. In addition, the speed of convergence of the split-beam phase retrieval algorithm is slower than the in-line case. Usually, 5 to 10 iterations will be enough for the in-line hologram's twin image elimination. However, as we can see from the Figure 6 that it needs around 70 to 100 iterations to have most of the twin image eliminated.

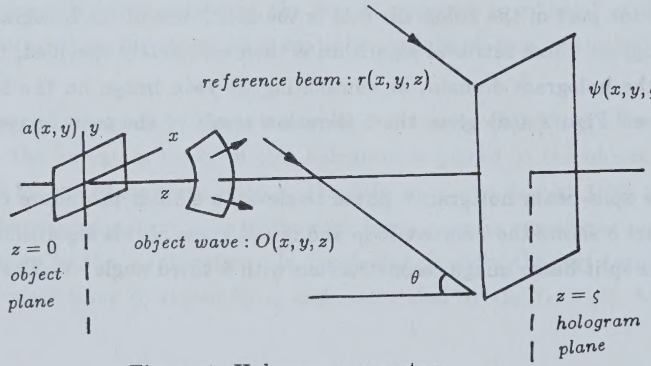


Figure 1. Hologram geometry

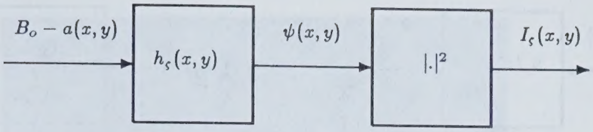

$B_o - a(x,y)$
$h_\varsigma(x,y)$
$\psi(x,y)$
$|.|^2$
$I_\varsigma(x,y)$

Figure 2. The in-line hologram formation procedure, B_o is a constant

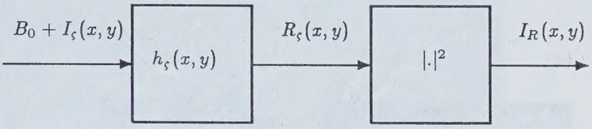

$B_0 + I_\varsigma(x,y)$
$h_\varsigma(x,y)$
$R_\varsigma(x,y)$
$|.|^2$
$I_R(x,y)$

Figure 3. The in-line hologram reconstruction procedure

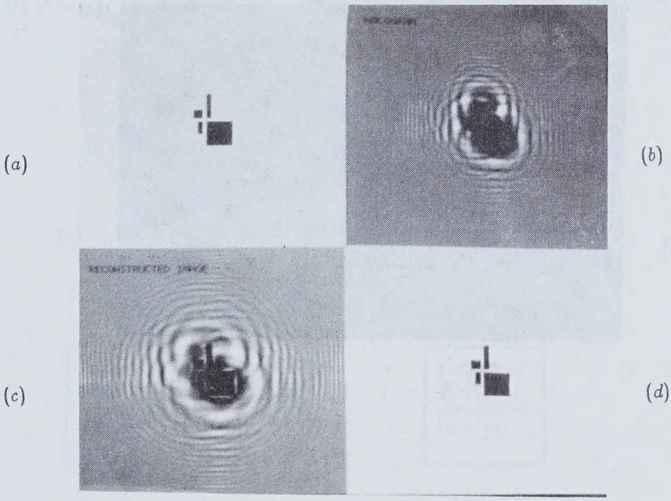

(a)
HOLOGRAM
(b)
RECONSTRUCTED IMAGE
(c)
(d)

Figure 4. (a), (b), and (c) are simulated object, hologram and reconstructed image for in-line hologram, (d) is the phase retrieval algorithm after 5 iterations

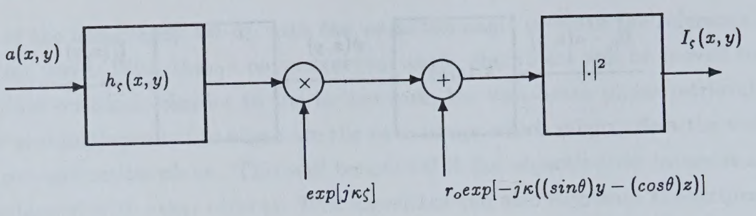
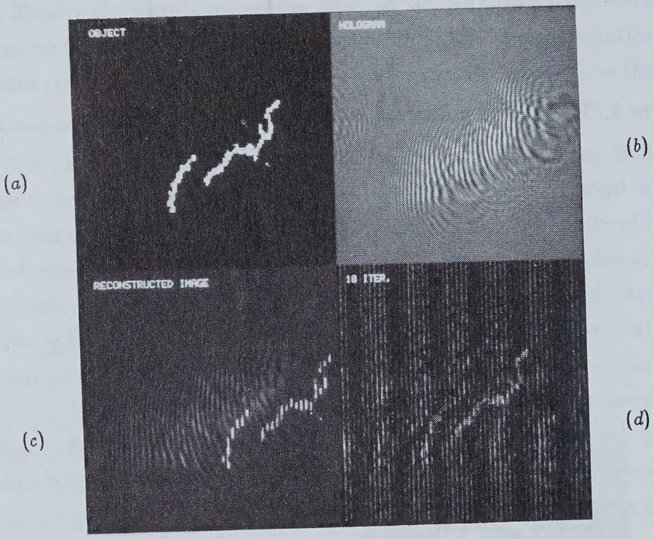

$a(x,y)$
$h_\varsigma(x,y)$
$\times$
$+$
$|.|^2$
$I_\varsigma(x,y)$
$exp[j\kappa\varsigma]$
$r_o exp[-j\kappa((sin\theta)y-(cos\theta)z)]$

Figure 5. The split-beam hologram formation procedure


OBJECT
HOLOGRAM
RECONSTRUCTED IMAGE
10 ITER.
(a)
(b)
(c)
(d)

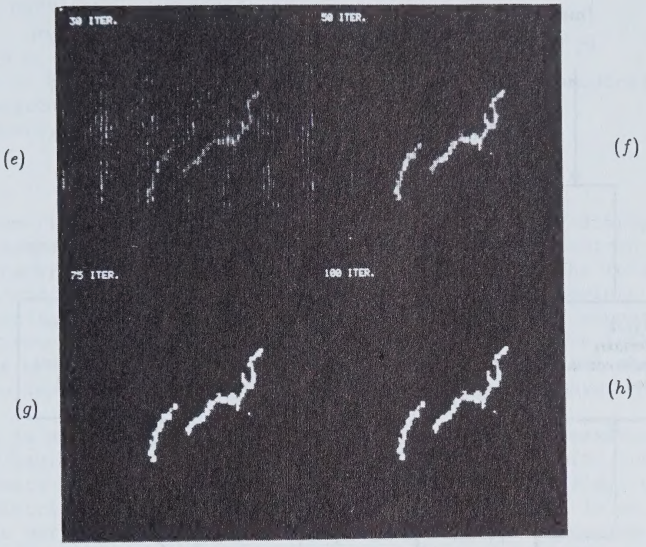

30 ITER.
50 ITER.
(e)
(f)
75 ITER.
100 ITER.
(g)
(h)

Figure 6.(a), (b) and (c) are split-beam hologram's object, hologram and reconstruction, (d), (e), (f), (g) and (h) are the results after 10, 30, 50, 75 and 100 iterations

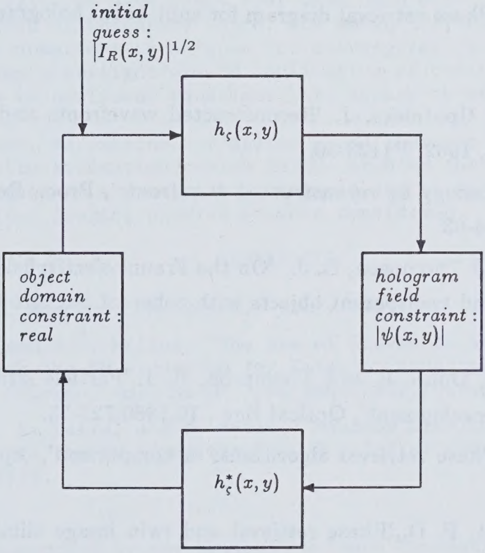

initial guess : $|I_R(x,y)|^{1/2}$
$h_\varsigma(x,y)$
hologram field constraint : $|\psi(x,y)|$
$h_\varsigma^*(x,y)$
object domain constraint : real

Figure 7. Phase retrieval diagram for in-line holography

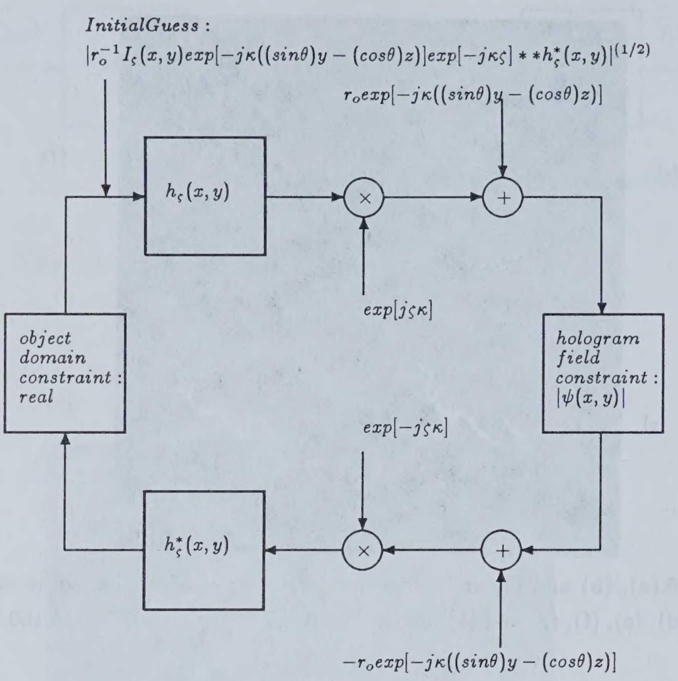

InitialGuess :
$|r_o^{-1} I_\varsigma(x,y) exp[-j\kappa((sin\theta)y-(cos\theta)z)] exp[-j\kappa\varsigma] ** h_\varsigma^*(x,y)|^{(1/2)}$
$r_o exp[-j\kappa((sin\theta)y-(cos\theta)z)]$
$h_\varsigma(x,y)$
×
+
$exp[j\varsigma\kappa]$
object domain constraint : real
hologram field constraint : $|\psi(x,y)|$
$exp[-j\varsigma\kappa]$
$h_\varsigma^*(x,y)$
×
+
$-r_o exp[-j\kappa((sin\theta)y-(cos\theta)z)]$

Figure 8. Phase retrieval diagram for split-beam holography

REFERENCES

1. Leith, E. N. and Upatnieks, J. 'Reconstructed wavefronts and communications theory', JOSA 52, 1962, P1123-30.
2. Gabor, D., 'Microscopy by reconstructed wavefronts', Proc. Royal Soc. London A, 197, 1949, p454-63.
3. Parrent, G. B. and Thompson, B. J., 'On the Fraunhofer(farfield) diffraction patterns of opaque and transparent objects with coherent background', Optica Acta , 11, 1964, 183-93.
4. Cartwright, S. L., Dunn, P. and Thompson, B. J.,'Particle sizing using far field holography:new development', Optical Eng., 19,1980,727-33.
5. Fienup, J. R.,' Phase retrieval algorithms: a comparison', Applied Optics, 21, 1982, 2758-68.
6. Liu, G. and Scott, P. D.,'Phase retrieval and twin image elimination for inline Fresnel holograms', to appear Jan. 1987, JOSA A.
7. Goodman, J. W., 'Introduction to Fourier optics', McGraw-Hill, 1968, p48-52.

THE USE OF GRENANDER'S SIEVES IN QUANTUM-LIMITED IMAGING*

DONALD L. SNYDER and MICHAEL I. MILLER
Dept. of Electrical Engineering and Institute for Biomedical Computing
Washington University
St. Louis, Missouri 63130

SUMMARY

Quantum-limited imaging problems occur in a number of disciplines, including gamma-ray astronomy, emission tomography, and electron-microscope autoradiography. Measured data in these problems are in the form of point locations, and the image to be formed from these measured points is the intensity function of another, underlying point process that cannot be observed directly. The locations of the measured points are influenced both by the underlying intensity function and by the physical characteristics of the instrumentation used to acquire the measured data.

There is a fundamental flaw in the way the method of maximum likelihood has been used to estimate the underlying intensity function. This manifests itself in the form of high peaks and low valleys seemingly randomly distributed throughout the images produced, which is an artifact that can be very pronounced and disturbing when seen. An explanation for this effect is that no constraints have been imposed in forming the maximum-likelihood estimate; adjacent pixels in the image are permitted to vary independently, and they do so with substantial variation.

We have been exploring the use of Grenander's sieves [1] as a method for introducing constraints into the maximum-likelihood method for quantum-limited imaging [2]. We have found that when used with the alternating maximization method of Dempster, Laird, and Rubin [3], this approach readily leads to numerical techniques for solving for the constrained maximizer, whereas a straightforward application of constrained maximum-likelihood leads to nonlinear equations that appear to be unsolvable.

In this paper, we outline our approach and indicate its application to probability density estimation, which is the problem that originally motivated the method of sieves to be developed and is closely related to the quantum-limited imaging problem we have considered.

REFERENCES

[1] U. Grenander, *Abstract Inference*, Wiley, 1981.

[2] D.L. Snyder and M.I. Miller, "The Use of Sieves to Stabilize Images Produced with the EM Algorithm for Emission Tomography," *IEEE Trans. on Nuclear Science*, vol. NS-32, pp. 3864-3872, October 1985.

[3] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Royal Statistical Soc.*, B, 39, pp. 1-37, 1977.

---

*This work was supported by the National Science Foundation under grant ECS-8407910 and by the National Institutes of Health under grant RR01380 from the Division of Research Resources.

NEW APPROACHES TO REAL-TIME TRANSFORM CODING AND RECONSTRUCTION OF SIGNALS AND IMAGES

O. K. Ersoy, C. H. Chen
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907

ABSTRACT

New approaches to transform coding and reconstruction of signals and images are discussed by developing new discrete Fourier-like transforms. The scrambled real discrete Fourier transform (SRDFT) has basically the same visual performance as the discrete cosine transform (DCT) in image compression, with much lower complexity of operations. Further generalizations of SRDFT and the scrambled DFT (SDFT) include all known fast discrete trigonometric transforms which diagonalize a number of important signal matrices. Fast Karhunen-Loeve transform (FKLT) is derived by following SRDFT with a number of Givens' plane rotations. It is also shown that the real and imaginary Fourier components should be treated independently when dealing with real signals and images. Both theoretical and experimental results are discussed.

1. INTRODUCTION

Transform coding of signals and images is one of the most efficient methods for signal and image compression. The purpose of the transform mapping is to reduce the correlation between signal elements and to concentrate the signal energy in the minimum number of transform coefficients.

The optimal transform which satisfies a number of criteria such as minimum mean square truncation error, uncorrelated transform coefficients and minimum entropy is the Karhunen-Loeve transform (KLT) [1]. Unfortunately, KLT is signal-dependent and difficult to implement in real-time. As a consequence, KLT is often approximated by discrete trigonometric transforms such as discrete Fourier (DFT), cosine (DCT) [2] and the generalized real discrete Fourier (GRDFT) [3] transforms.

DCT is generally recognized to be the best approximation to KLT, especially for Markov-1 type signals [4]. Another advantage of DCT is the minimum presence of edge-effect (blocking) artifacts when the signal or image is treated in blocks.

An important consideration in transform image compression is the suitability to real-time and VLSI implementations. Simple transforms such as the Hadamard transform are more advantageous in terms of implementation, but they do not achieve the same degree of compression [5].

In this article, a number of new fast discrete trigonometric transforms will be developed, achieving the best approximation to KLT in the least number of operations, as well as satisfying other criteria of reconstruction such as minimal edge effects. Experiments using the new transforms on signals and images will be discussed, in comparison with previously known transforms.

2. THE SCRAMBLED REAL DISCRETE FOURIER TRANSFORM

RDFT can be written as [6]

$$y(n) = \sum_{k=0}^{N-1} x(k)\cos\left[\frac{2\pi nk}{N} + \theta(n)\right] \tag{2.1}$$

where

$$\theta(n) = 0 \quad 0 \le n \le N_1 \tag{2.2}$$

$$= \frac{\pi}{2} \quad N_1 < n < N \tag{2.2}$$

and N_1 equals N/2 when N is even, (N-1)/2 when N is odd.

The inverse transform is given by

$$x(n) = \frac{2}{N} \sum_{k=0}^{N-1} y(k)v(k)\cos\left[\frac{2\pi nk}{N} + \theta(k)\right] \tag{2.3}$$

where

$$\begin{aligned} v(k) &= 1 \qquad k \neq 0, N/2 \\ &= \frac{1}{2} \qquad k = 0, N/2 \end{aligned} \tag{2.4}$$

DCT can be written as [7]

$$X_c(n) = \sum_{k=0}^{N_1-1} x(k)\cos \frac{2\pi n(k+1/2)}{N} \tag{2.5}$$

where $n = 0,1,2...(N_1-1)$.

The inverse transform is

$$x(n) = \frac{4}{N} \sum_{k=0}^{N_1-1} X_c(k)\cos \frac{2\pi k(n+1/2)}{N} \tag{2.6}$$

It can be shown that [8] and z(·) is defined by

$$X_c(n) = \sum_{k=0}^{N_1-1} z(k)\cos\left[\frac{2\pi nk}{N_1} + \phi(n)\right] \tag{2.7}$$

where

$$\left.\begin{aligned} z(k) &= x(2k) \\ z(N_1-1-k) &= x(2k+1) \end{aligned}\right\} \quad k = 0,1,...(N_1/2) \tag{2.8}$$

$$\begin{aligned} \phi(n) &= \frac{\pi n}{N} \qquad 0 \le n \le \frac{N_1}{2} \\ &= \frac{\pi n}{N} + \frac{\pi}{2} \quad \frac{N_1}{2} < n < N_1 \end{aligned} \tag{2.9}$$

There are 4 classes of GRDFT [3]. (2.10) is the same as GRDFT of class I, or $R_1(\phi)$, which is the generalization of (2.1) with a phase angle $\phi(\cdot)$. Thus, DCT is the same as $R_1(\phi)$ of the permuted input data, permutations being given by (2.8). By the same token, RDFT is $R_1(0)$.

SRDFT will be defined as $R_1(0)$ of z(·). It can be written as

$$y(n) = \sum_{k=0}^{N-1} x(k)\cos\left[\frac{2\pi nk'}{N} + \theta(n)\right] \tag{2.10}$$

where

$$\begin{aligned} k' &= k/2 \qquad k \text{ even} \\ &= -(k+1)/2 \quad k \text{ odd} \end{aligned} \tag{2.11}$$

The inverse SRDFT is

$$x(k) = \frac{2}{N} \sum_{n=0}^{N-1} y(n)v(n)\cos\left[\frac{2\pi nk'}{N} + \theta(n)\right] \tag{2.12}$$

2-D SRDFT with dimensions N_1 and N_2 can be written as

$$y(n_1,n_2) = \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} x(k_1,k_2)\cos\left[\frac{2\pi n_1 k_1'}{N_1} + \theta(n_1)\right]\cos\left[\frac{2\pi n_2 k_2'}{N_2} + \theta(n_2)\right] \qquad (2.13)$$

with the inverse transform

$$x(k_1,k_2) = \frac{4}{N_1 N_2}\sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} y(n_1,n_2)\cos\left[\frac{2\pi n_1 k_1'}{N_1} + \theta(n_1)\right]\cos\left[\frac{2\pi n_2 k_2'}{N_2} + \theta(n_2)\right] \qquad (2.14)$$

k_1' and k_2' in (2.13) and (2.14) are defined as in (2.11).

3. THE COMPLEXITY OF SRDFT VERSUS DCT

The complexity of SRDFT in terms of number of additions A(N), and multiplications M(N) is given by [8]

$$\begin{aligned} A(N) &= \frac{N}{2}(3\log_2 N - 5) + 4 \\ M(N) &= \frac{N}{2}(\log_2 N - 3) + 2 \end{aligned} \qquad (3.1)$$

DCT is the same as SRDFT followed by Givens' plane rotations due to $\phi(\cdot)$ in (2.10). The optimal implementation of a Givens' plane rotation is in terms of 3 multiplications and 3 additions.

The corresponding number of operations with 2-D data of dimensions N_1 by N_2 are $N_1 O(N_2) + N_2 O(N_1)$, where $O(\cdot)$ indicates operation count (of additions or multiplications) and transformations are done first rowwise and then columnwise (or vice versa). Assuming N_1, N_2 to be 512, and the 2-D data to be processed in blocks of dimensions M by M, the resulting number of operations are shown in Table 1. Especially with small block sizes, Table 1 shows that SRDFT is very advantageous as compared to DCT in terms of number of operations.

4. MEASURES OF PERFORMANCE

In order to investigate the performance of the new transform in coding and filtering, Pearl's measure $|T-T_U|$ can be used for a transform with matrix U; T is the covariance matrix of the signal, and T_U is given by [9]

$$T_U = U^{-1} D\, U \qquad (4.1)$$

$$T_U' = U T U^{-1} \qquad (4.2)$$

$$D_U = \mathrm{diag}\, T_U' \qquad (4.3)$$

$|T-T_u|^2$ is the Hilbert-Schmidt norm of $T-T_U$ given by

$$|T-T_U|^2 = \frac{1}{N}\left[\sum_{m,n=0}^{N-1} |T-T_U|_{mn}^2\right] \qquad (4.4)$$

which can also be written as

$$|T-T_U|^2 = |T_U' - D_U|^2 = |T|^2 - \frac{1}{N}\sum_{n=0}^{N-1} |UTU^{-1})_{nn}|^2 \qquad (4.5)$$

The normalized correlation measure Q is defined by

$$Q = \frac{|T-T_U|^2}{|T-I|^2} \qquad (4.6)$$

where I is the identity matrix. Q represents the fractional correlation left "undone" by the transformation.

Markov-1 signal models with a single parameter ρ, $0 < p < 1$, represent images quite adequately [5].

For $\rho = .95$ (a typical value for images), $Q = .00598$ with DCT, and .00681 with SRDFT. Thus, the performance of the two transforms are very similar with the Markov-1 signal model. The same conclusion is reached in the next section in terms of the reconstruction error with actual images.

Another major issue limiting performance is the edge-effect artifacts. The performance of DCT in this respect will be analyzed first.

DCT of size N_1 can be written as [10]

$$X_c(n) = e^{-\frac{j\pi n}{N}} X_F(n) \tag{4.7}$$

where

$$X_F(n) = \sum_{k=0}^{N-1} x'(k) e^{-\frac{j2\pi nk}{N}} \tag{4.8}$$

and $x'(k)$ is given by

$$\begin{aligned} x'(k) &= x(k) & k &= 0,1,\ldots,N_1-1 \\ &= x(N-1-k) & k &= N_1, N_1+1, \ldots, N-1 \end{aligned} \tag{4.9}$$

$x'(\cdot)$ is symmetric about a point one-half sample between $k = N_1\text{-}1$ and $k = N_1$. The main reason why DCT results in minimal edge-effect artifacts is that it is related to DFT of $x'(\cdot)$ according to (4.7), and $x'(\cdot)$ has no jump discontinuities as a periodic signal, thereby avoiding Gibbs phenomenon.

SRDFT will be shown to be similarly related to DFT of $x'(\cdot)$. Let C and R_s represent the DCT and SRDFT matrices. Using (2.7), the relationship between C and R_s of size N_1 can be shown to be

$$R_s = Z\,C \tag{4.10}$$

where the matrix Z has elements $Z_{m,n}$ given by

$$\begin{gathered} Z_{0,0} = 1 \\ Z_{\frac{N_1}{2},\frac{N_1}{2}} = 1 \quad \text{if } N_1 \text{ even} \\ \left.\begin{aligned} Z_{n,n} &= Z_{N_1-n,N_1-n} = \cos\frac{\pi n}{N} \\ Z_{n,N_1-n} &= -Z_{N_1-n,n} = \sin\frac{\pi n}{N} \end{aligned}\right\} \; 0 < n < \frac{N_1}{2} \\ Z_{m,n} = 0 \quad \text{otherwise} \end{gathered} \tag{4.11}$$

Using (4.7) and (4.10), the relationship between $y(\cdot)$ of SRDFT and $X_F(\cdot)$ can be shown to be

$$y(0) = X_F(0)$$

$$y\left(\frac{N_1}{2}\right) = X_F\left(\frac{N_1}{2}\right) \text{ if } N_1 \text{ even}$$

$$\left.\begin{aligned} y(n) &= \mathrm{Re}[X_F(n) - e^{j\frac{\pi n}{N}} X_F(N_1-n)] \\ y(N_1-n) &= \mathrm{Im}[X_F(n) - e^{-j\frac{\pi n}{N}} X_F(N_1-n)] \end{aligned}\right\} \; 0 < n < \frac{N_1}{2} \tag{4.12}$$

Thus, $y(\cdot)$ is related to $X_F(\cdot)$ by (4.12), $X_F(\cdot)$ being the DFT of $x'(\cdot)$, a signal without jump discontinuities at $n = N_1$. As a consequence, SRDFT can be expected to result in minimal edge-effect artifacts, just like DCT. This conclusion will also be supported by experimental evidence in the next section.

5. EXPERIMENTS

The image to be experimented with is shown in Fig. 1. It is of size 512 by 512. Processing was done in blocks of 16 by 16, both with DCT and SRDFT. After transformation, 16 elements of largest variance were chosen out of a total of 256 elements, the other elements being set to zero. Then the image was reconstructed.

The reconstructed images are shown in Figs. 2 and 3. The reconstruction error defined as the mean square root error per pixel divided by the average amplitude was .0761 (DCT), .0798 (SRDFT). The visual quality of the reconstructions were basically the same with both methods. In addition, SRDFT achieves a reduction of 688128 multiplications and 688128 additions.

In order to determine the effectiveness of SRDFT in reducing edge effects, the experiments were repeated with RDFT, thereby skipping the permutations given by (2.9). Fig. 4 is the RDFT reconstructions. The block edges are easily visible whereas they are invisible in Fig. 3. Thus, it is clear that SRDFT is very effective in minimizing edge-effect artifacts.

6. GENERALIZED SRDFT AND SDFT

When signals are complex, it may be more appropriate to use the scrambled DFT (SDFT), defined as follows:

$$y_F(n) = \sum_{k=0}^{N_1-1} x(k)\, e^{-2\pi j \left(\frac{nk'}{N_1} + \frac{n}{2N}\right)} \tag{6.1}$$

where $k' = 0,1,2,\ldots,(N_1\text{-}1)$ and k' is given by (2.11). $y_F(n)$ is the same as $X_F(n)$ of $x'(\cdot)$ defined by (4.9) for the first N_1 coefficients. As a consequence, SDFT should also result in minimal edge-effect artifacts for complex signals.

SRDFT and SDFT can be further generalized as follows:

SRDFT:

$$y_R(n) = \sum_{k=0}^{N-1} x(k) \cos\left(\frac{2\pi n' k'}{N} + \theta(n') + \phi(n)\right) \tag{6.2}$$

where $\theta(n')$ is given by (2.11); $\phi(n)$ is an arbitrary phases angle satisfying $\phi(n) = -\phi(N-n)$, $n \neq 0, \frac{N}{2}$; n' and k' are isomorphic permutations of the sequences of n and k, respectively; if $n' = p(n)$, it is also true that $N\text{-}n' = p(N\text{-}n)$.

The inverse transform is

$$x(k) = \frac{2}{N} \sum_{n=0}^{N-1} y_R(n) v(n) \cos\left(\frac{2\pi n' k'}{N} + \theta(n') + \phi(n)\right) \tag{6.3}$$

SDFT:

$$y_F(n) = \sum_{k=0}^{N-1} x(k)\, e^{-2\pi j \left(\frac{n'k'}{N} + \phi(n)\right)} \tag{6.4}$$

with the inverse transform

$$x(k) = \frac{1}{N} \sum_{n=0}^{N-1} y_F(n)\, e^{-2\pi j \left(\frac{n'k'}{N} + \phi(n)\right)} \tag{6.5}$$

All known discrete trigonometric transforms which diagonalize certain important matrices such as circular and skew-circular convolution, correlation matrices, tridiagonal matrices and some covariance matrices are special cases of SRDFT and SDFT as defined by (6.2) and (6.4).

Some of these matrices with elements t(i,j) are outlined below:

A. Circular correlation matrix given by t(i,j)=h((i+j) modulo N) with N independent elements h(0), h(1),..,h(N-1).

B. Circular convolution matrix given by t(i,j)=h((i-j) modulo N).

C. Tridiagonal matrix given by $t(i,j) = 1$, $t(i,i+1) = t(i,|i-1|) = -\alpha$.

D. Tridiagonal matrix same as in C, with the modifications $t(0,0) = t(N-1, N-1) = 1-\alpha$.

E. Tridiagonal matrix given by $t(i,i) = 1 + \alpha^2$, $t(1,2) = t(2,1) = t(N-2, N-1) = t(N-1, N-2) = \sqrt{2}\alpha$, $t(i,|i-1|) = -\alpha$.

F. Skew-circular correlation matrix given by t(i,j)=sign(N-i-j-1)h((i+j) mod N).

G. Skew-circular convolution matrix given by t(i,j)=sign(i−j)h((i−j)moduloN)

Table 2 shows the specific SRDFT's and SDFT's corresponding to these cases.

7. FAST KARHUNEN-LOEVE TRANSFORM (FKLT)

SRDFT as defined by (6.2) is expected to be a good approximation to KLT for many types of signals. Further improvement can be achieved by including a number of Givens' plane rotations following SRDFT.

The new covariance matrix T′ in the SRDFT domain is given by

$$T' = R_S^T\, T\, R_S \tag{7.1}$$

Since T is symmetric, so is T′. The largest off-diagonal elements of T′ can be reduced to 0, using Jacobi's method. For example, if the largest off-diagonal element of T′ is t_{jk}, the angle θ(i,j) of plane rotation to reduce t'_{jk} to zero is given by

$$\cot 2\theta(j,k) = \frac{t'_{jj} - t'_{kk}}{2t'_{jk}} \tag{7.2}$$

Successive plane rotations result in the diagonalization of T. Convergence is known to be quadratic. Let M be the total number of plane rotations to achieve a desired degree of accuracy. Fast KLT (FKLT) will be defined as SRDFT followed by M plane rotations. Let the nth plane rotation involve the indices j(n) and k(n). FKLT is given by

$$y_k(n) = \sum_{m,j(m)=n} y_R(n) \cos\theta(n,k(m)) + \operatorname{sign}(n-k(m))\, y_R(k(m)) \sin\theta(n,k(m)) \tag{7.3}$$

where $y_R(\cdot)$ is given by (6.2). The inverse FKLT is obtained by first finding

$$y_R(n) = \frac{1}{2^{J(n)}} \sum_{m,j(m)=n} y_k(n) \cos\theta(n,k(m)) - \operatorname{sign}(n-k(m))\, y_k(k(m)) \sin\theta(n,k(m)) \tag{7.4}$$

and then finding $x(\cdot)$ from $y_R(\cdot)$, using (6.2). J(n) in (7.4) is the number of times j(m) equals n.

Experiments were carried out in connection with Markov-1 signals in order to show the effectiveness of FKLT. The results were compared to DCT. FKLT was achieved by following SRDFT given by (2.12) with Givens' plane rotations. In order to obtain acceptable results over a given interval of ρ say (ρ_1,ρ_2), an average covariance matrix T_a was computed.

Fig. 5 shows Q versus ρ for DCT, FKLT (8 rotations) and FKLT with 1 rotation, when N is 16 and (ρ_1,ρ_2) =(.92,.97), an interval suitable for image processing. It is seen that FKLT has better performance than DCT in an interval much larger than the chosen (ρ_1,ρ_2). At $\rho = .95$, FKLT has negligible error. It is also seen that FKLT with one rotation performs very well.

Table 4 shows the rotation angle θ_m, position of mth rotation j,k and the off-diagonal element of the covariance matrix zeroed. Those values of m with an asterisk indicate those rotations also covered by DCT. It is seen that DCT misses 5 of the 8 largest off-diagonal elements.

Fig. 6 is the same as Fig. 1 except that (ρ_1,ρ_2) is chosen to be (.43, .48). It is seen that FKLT is better than DCT for low values of ρ as well, with only a few rotations.

8. A WORD OF CAUTION

Fourier transform is an extremely powerful tool to help solve diverse technical problems. Because of the ease the complex exponential provides as well as the power of complex analysis which can be put to use, the Fourier transform has almost always been formulated as a complex transform, thereby having conditioned us to think in terms of amplitude and phase.

When we code digital signals with the discrete Fourier transform, we would naturally consider the terms with the highest amplitude, setting the smaller ones to zero. If we use the real discrete Fourier transform instead, we can consider each transform component independent of others, and choose the ones with the largest absolute value to code the signal. The second approach turns out to be more optimal.

Let us call the 2 methods above method I and II, respectively. Figs. 7 and 8 show a signal of size 128, and its reconstructions according to the 2 methods, after having kept 16 largest amplitude terms according to method I, and 16 largest absolute value coefficients according to method II. Method II is obviously better. The mean square errors are 2.42779 (method I) and 1.00347 (method II). We have studied a number of other signals as well, always reaching the conclusion that the real and the imaginary Fourier coefficients of real signals should be treated independently of each other for optimal coding and reconstruction of signals.

REFERENCES

1. K. Fukunaga, *Introduction to Statistical Pattern Recognition,* Academic Press, 1972, pp. 226-254, Academic Press.
2. H. B. Kekre, J. K. Solanki, "Comparative Performance of Various Trigonometric Unitary Transforms for Transform Image Coding," *Int. J. Electron.,* Vol. 44, 305-315, 1978.
3. O. K. Ersoy, "Generalized Real Discrete Fourier Transform: A Family of Transforms," *XX. Princeton Conference on Information Sciences and Systems,* pp. 777-782, March 1986.
4. M. Hamidi, J. Pearl, "Comparison of Cosine and Fourier Transforms of Markov-1 Signals," *IEEE Trans. Acoustics, Speech, Signal Processing,* Vol. ASSP-24, 428-429, 1976.
5. A. N. Netravalli, J. O. Limb, "Picture Coding: A Review," *IEEE Proceedings,* Vol. 68, No. 3, 366-406, March 1980.

6. O.K. Ersoy, "Real Discrete Fourier Transform," *IEEE Trans. Acoustics, Speech, Signal Processing,* Vol. ASSP-33, No. 4, 880-882, August 1985.

7. N. Ahmed, T. Natarajan, K. R. Rao, "Discrete Cosine Transform," *IEEE Trans. Computers,* Vol. 23, No. 1, 90-93, January 1974.

8. O. K. Ersoy, N. C. Hu, "A Unified Approach to the Fast Computation of All Discrete Trigonometric Transforms," *Purdue University Technical Report,* No. TR-EE 86-32, September 1986, and submitted to *IEEE Trans. Acoustics, Speech, Signal Processing.*

9. J. Pearl, "On Coding and Filtering Stationary Signals by Discrete Fourier Transforms," *IEEE Trans. Acoustic, Speech, Signal Processing,* Vol. IT-19, 229-232, March 1973.

10. W. Chen, S.C. Fralick, "Image Enhancement Using Cosine Transform Filtering," *Proc. Symp. Current Math. Probl. Image Sci.*, pp. 186-192, Monterey, California, 1976.

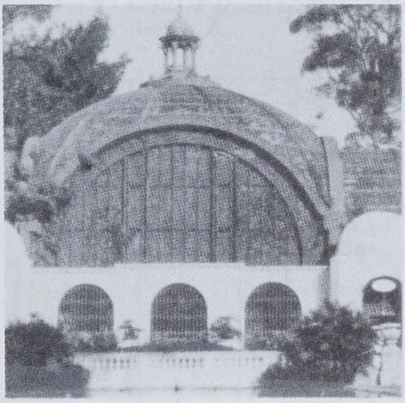


Fig. 1. The Original Image of Size 512 by 512.

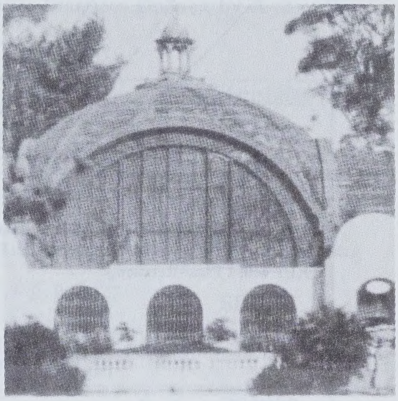


Fig. 2. The Reconstructed Image with DCT.

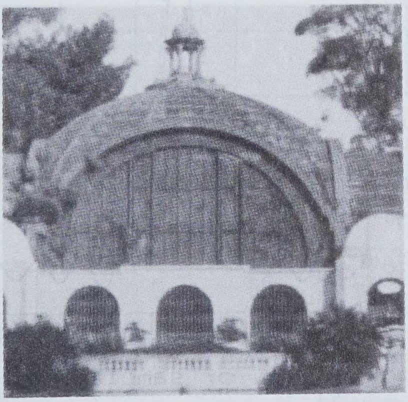


Fig. 3. The Reconstructed Image with SRDFT.

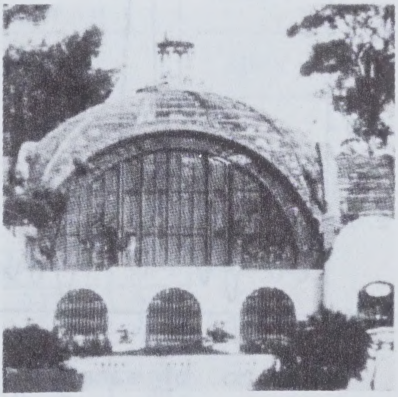


Fig. 4. The Reconstructed Image with RDFT

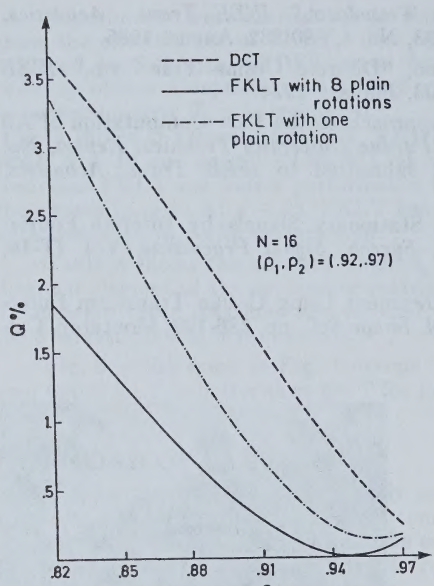


Fig. 5. The Normalized Correlation Measures for DCT, FKLT with 8 Plane Rotations, and FKLT with 1 Plane Rotation

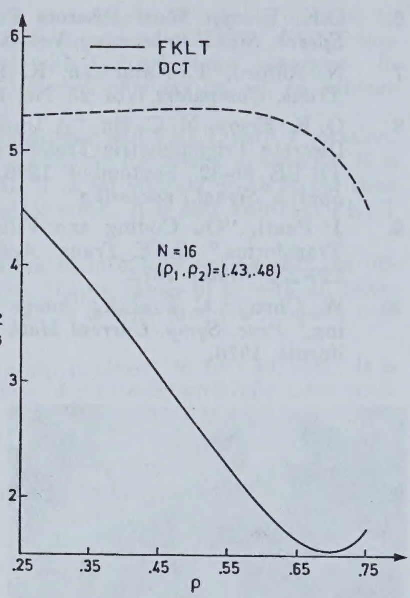


Fig. 6. The Normalized Correlation Measures for DCT and FKLT at $(\rho_1,\rho_2) = (.43,.48)$.

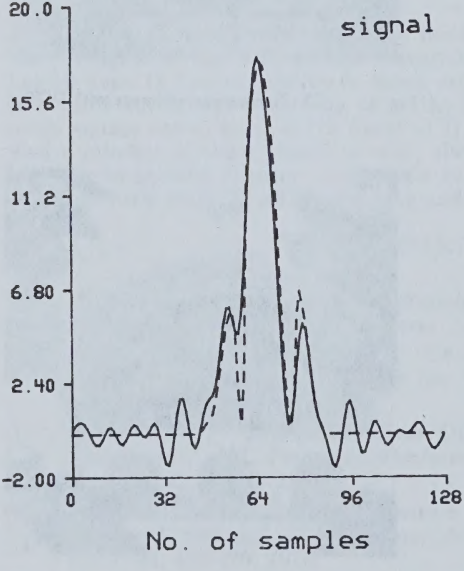


Fig. 7. The Original Signal (Dashed line) and Its Reconstruction (Solid Line) by choosing 16 Largest Absolute Value Coefficients of RDFT.

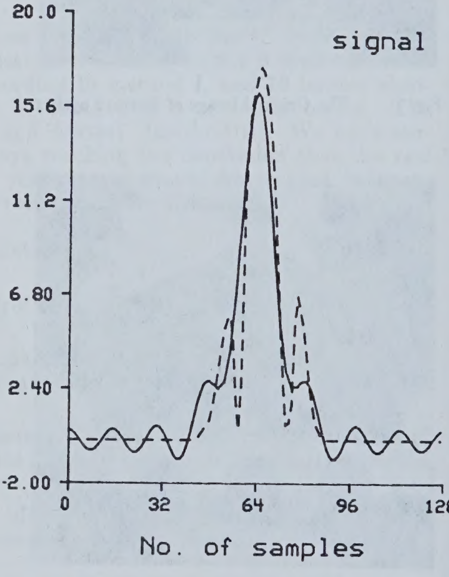


Fig. 8. The Original Signal and Its Reconstruction by Choosing 16 Largest Amplitude Coefficients of DFT.

Table 1. The Number of Additions and Multiplications in the Computation of 2-D SRDFT and DCT in Blocks of Dimensions M by M When the Total 2-D Data Size is 512 by 512.

	Additions		Multiplications	
M	SRDFT	DCT	SRDFT	DCT
4	786,432	1,179,648	0	524,288
8	1,310,720	1,900,544	131,072	786,432
16	1,966,080	2,654,208	327,680	1,048,576
32	2,686,976	3,424,256	557,056	1,310,720
64	3,440,640	4,202,496	802,816	1,572,864
128	4,210,688	4,984,832	1,056,768	1,835,008
256	4,988,928	5,769,216	1,314,816	2,097,152
512	5,771,264	6,554,624	1,574,912	2,359,296

Table 2. GSRDFT and GSDFT which Diagonalize Matrices Defined by A thru G in Section 6.

Case	n′	k′	$\phi(n)$	Transform
A	n	k	$\frac{1}{2}\cot^{-1}(\frac{y_R(n)^*}{y_R(N-n)})$	SRDFT
B	n	k	0	SDFT
C	n	k	0	SRDFT (sine part)
D	n	$\frac{k}{2}$,k even $-\frac{k+1}{2}$,k odd	$\frac{\pi n}{N}, 0 \leq n \leq \frac{N}{2}$ $\frac{\pi n}{N} + \frac{\pi}{2}, \frac{N_1}{2} < n < N_1$	SRDFT (cosine part
E	n	k	0	SRDFT (cosine part)
F	$n + \frac{1}{2}$	k	$\frac{1}{2}\cot^{-1}(\frac{y_R(N-n)}{y_R(n)})$	SRDFT
G	$n = \frac{1}{2}$	k	0	SDFT

* $y_R(\cdot)$ is RDFT coefficients.

Table 3. Rotation angle θ_m position of mth rotation j,k and the off-diagonal element of covariance matrix t(j,k) zeroed.

m	j	k	θ_m(degrees)	t(j,k)
1	1	3	3.139	-5.06
2*	2	10	5.644	-1.56
3	1	5	.729	-1.22
4	1	11	-.606	1.02
5*	3	11	-10.229	.98
6*	4	12	16.927	-.65
7	2	4	2.595	-.62
8	1	7	-.324	.55

SPECTRAL PROPERTIES OF A FINITE FIELD DFT

M. W. MARCELLIN AND T. R. FISCHER
Department of Electrical Engineering
Texas A & M University
College Station, TX 77843

Finite field arithmetic has been proposed for a number of signal processing applications [1], with perhaps fast finite impulse response filtering and residue number system (RNS) processing the most important. One advantage of RNS processing is that the algebraic integers may be used to approximate an arbitrary complex number with a bit accuracy requirement that is far less than that of straightforward (Gaussian integer) approximation [2]. Such algebraic integers may be thought of as certain coefficients of the standard complex field discrete Fourier transform (DFT) of an all integer sequence. Finite field arithmetic and the finite field DFT (FFDFT), the complex field DFT, and the RNS approach to signal processing are related, and it is of value to characterize this relationship. This paper focuses on the relationship between FFDFT coefficients and (complex field) DFT coefficients (for a common integer data sequence) and the information required to convert from one set of transform coefficients to the other. For simplicity, the discussion is restricted to the radix 2 DFT and the Fermat fields $\mathrm{GF}(2^l+1)$ where l is chosen such that 2^l+1 is prime [1].

Let $\mathbf{x}$ be an N-tuple with $N \in \{2,4,8,\ldots,2^l\}$ and $x_n \in \mathrm{GF}(2^l+1)$ for $n = 0,1,\ldots,N-1$ and let $W_N \in \mathrm{GF}(2^l+1)$ be an element of order N. Then the FFDFT of $\mathbf{x}$, denoted $\mathbf{X}_F$, is defined by $X_{F_k} = \sum_{n=0}^{N-1} x_n W_N^{nk}$ for $k = 0,1,\ldots,N-1$ where all operations are carried out modulo (2^l+1) (i.e. within the field). Clearly x_n is also an element of the complex field $\mathbf{C}$. Hence, there exists a standard complex field DFT of $\mathbf{x}$, say $\mathbf{X}$, where $X_k = \sum_{n=0}^{N-1} x_n e^{-j\frac{2\pi nk}{N}}$ for $k = 0,1,\ldots,N-1$.

The phasor $e^{-j\frac{2\pi}{N}}$ has well known conjugate and negative symmetries, so that X_k can be written as $\sum_{m=0}^{\frac{N}{4}-1} a_{k_m} e^{-j\frac{2\pi m}{N}}$ where each $a_{k_m} \in \mathbf{Z}$ (the integers) and is an appropriate combination of the x_n's. Since each exponential can be expressed as $\cos(\frac{2\pi m}{N}) - j\sin(\frac{2\pi m}{N})$, and $\sin(\frac{2\pi m}{N}) = \cos(\frac{\pi}{2} - \frac{2\pi m}{N})$, $X_k = \sum_{m=0}^{\frac{N}{4}-1}(b_{k_m} + jc_{k_m})\cos(\frac{2\pi m}{N})$ for some b_{k_m}'s, c_{k_m}'s $\in \mathbf{Z}$. After examining the decimation in frequency FFT algorithm, it is clear that for k even, many of the b's and c's are zero. In fact, a relatively simple induction argument will show that

$$X_k = \sum_{m=0}^{M-1}(b_{k_m} + jc_{k_m})\cos\left(\frac{2\pi m}{4M}\right), M = \begin{cases} \frac{N}{4}, & k = 1,3,5,\ldots,N-1; \\ \frac{N}{8}, & k = 1(2),3(2),\ldots,(\frac{N}{2}-1)(2); \\ \vdots & \vdots \\ 2, & k = 1(\frac{N}{8}),3(\frac{N}{8}),5(\frac{N}{8}),7(\frac{N}{8}); \\ 1, & k = 1(\frac{N}{4}),3(\frac{N}{4}). \end{cases} \tag{1}$$

Note that X_0 and $X_{\frac{N}{2}}$ are single integers.

Now we wish to relate $\mathbf{X}$ to $\mathbf{X}_F$. The obvious relation is $\mathbf{X} \xrightarrow{IDFT} \mathbf{x} \xrightarrow{FFDFT} \mathbf{X}_F$, but (1) suggests that a computationally simpler relation exists. For $M = 1,2,\ldots,\frac{N}{4}$, $\cos(\frac{2\pi m}{4M}) = \cos(\frac{2\pi m}{4}), \cos(\frac{2\pi m}{8}),\ldots,\cos(\frac{2\pi m}{N})$. Also, as m varies, $|\cos(\frac{2\pi m}{4})| \in \{0,1\}$, $|\cos(\frac{2\pi m}{8})| \in \{0,1,\sqrt{2}/2\}$, $|\cos(\frac{2\pi m}{16})| \in \{0,1,\sqrt{2}/2,\sqrt{\frac{1+\sqrt{2}/2}{2}},\sqrt{\frac{1-\sqrt{2}/2}{2}}\}$, etc. Each of these quantities ($\sqrt{2}/2, \sqrt{\frac{1+\sqrt{2}/2}{2}}$, etc.) exists in GF($2^l+1$), provided $N \in \{2, 4, \ldots, 2^l\}$. Furthermore, $\cos(\frac{2\pi}{N}) - j\sin(\frac{2\pi}{N}) = \cos(\frac{2\pi}{N}) - j\cos(\frac{\pi}{2} - \frac{2\pi}{N})$ induces an element of order N in GF(2^l+1) when j is interpreted as $\sqrt{-1} = \sqrt{2^l} = 2^{l/2}$. Note that $l/2 \in \mathbf{Z}$ since $2^l + 1$ is known to be composite for l not a power of 2.

Example: Let $l = 8$, then $x_n \in$ GF(257). $\cos(\frac{2\pi}{2}) = -1 \rightarrow 256 \in$ GF(257), $\sin(\frac{2\pi}{2}) = 0$ which yields $W_2 = 256$. $\cos(\frac{2\pi}{4}) = 0$, $\sin(\frac{2\pi}{4}) = 1$ which yields $W_4 = -16 = 241$. $\cos(\frac{2\pi}{8}) = \sin(\frac{2\pi}{8}) = \sqrt{2}/2 \rightarrow \pm 30 \in$ GF(257). Arbitrarily choosing 30 yields $W_8 = 30 - 16(30) = 64$. Now, $\cos(\frac{2\pi}{16}) = \sqrt{\frac{1+\sqrt{2}/2}{2}} \rightarrow \sqrt{\frac{1+30}{2}} = \pm 12 \in$ GF(257) and $\sin(\frac{2\pi}{16}) = \sqrt{\frac{1-\sqrt{2}/2}{2}} \rightarrow \pm 63 \in$ GF(257). If 12 is chosen for $\cos(\frac{2\pi}{16})$ then $-63 = 194$ must be chosen for $\sin(\frac{2\pi}{16})$ so that $W_{16} = 12 - 16(194) = 249$ is a square root of W_8. Proceeding in this manner, the following table can be constructed.

N	2	4	8	16	32	64	128	256
$\cos(\frac{2\pi}{N})$	256	0	30	12	69	158	145	79
$\sin(\frac{2\pi}{N})$	0	1	30	194	61	123	214	151
W_N	256	241	64	249	121	246	62	233

For the sake of brevity, only values corresponding to $\cos(\frac{2\pi}{N})$ and $\sin(\frac{2\pi}{N})$ have been listed. Following a similar procedure, $\cos(\frac{2\pi m}{N})$ and $\sin(\frac{2\pi m}{N})$ can be tabulated for $m = 0, 1, \ldots, N-1$.

One application of the previous discussion is that given $X_k \in \mathbf{C}$, $X_{F_k} \in$ GF(2^l+1) can be found by simply substituting the values from GF(2^l+1) that correspond to $\cos(\frac{2\pi m}{N})$ into the expression $X_k = \sum_{m=0}^{M-1}(b_{k_m} + jc_{k_m})\cos(\frac{2\pi m}{4M})$ and carrying out all operations within the field.

Another application is that a finite field discrete cosine transform (FFDCT) can be defined. The discrete cosine transform is defined by $X_k = \sqrt{\frac{2}{N}}\alpha_k \sum_{n=0}^{N-1} x_n \cos\frac{(2n+1)k\pi}{2N}$, $k = 0, 1, \ldots, N-1$ where $\alpha_0 = 1/\sqrt{2}$ and $\alpha_k = 1$, $k \neq 0$ [3]. Note that for $N = 2, 4, 8, \ldots, 2^{l-2}$, $\cos\frac{(2n+1)k\pi}{2N}$ has been defined in GF(2^l+1). Substituting and carrying out all operations modulo 2^l+1, a transform that is both unitary and invertible is obtained. It can also be shown that the popular fast DCT algorithm given by $X_k = \sqrt{\frac{1}{2N}}\alpha_k W_{4N}^k X_k'$ where X_k' is the k^{th} element from the Fourier transform of the $2N$ length even extension of $\mathbf{x}$ can be used to implement the FFDCT defined above.

Acknowledgement

This work was supported, in part, by a General Electric Fellowship.

References

[1] R. E. Blahut, "Algebraic Fields, Signal Processing, and Error Control," *Proc. IEEE*, vol. 73, no. 5, pp. 874-893, 1985.

[2] R. A. Games, "Complex Approximations Using Algebraic Integers," *IEEE Trans. Inform. Theory*, vol. IT-31, no. 5, pp. 565-579, 1985.

[3] N. S. Jayant and P. Noll, *Digital Coding of Waveforms*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1984.

Robust Detectors for Very Heavy-Tailed Noise

Douglas Warren
University of Illinois at Chicago
Chicago, IL 60607
and
John B. Thomas
Princeton University
Princeton, NJ 08544

Abstract

This paper treats the robust detection problem for the case where the noise mixture model is not strongly unimodal; that is, has heavier than exponential tails. Under a constraint on the support of the contaminating density, an optimal detector is derived. Its performance is investigated using both computer-generated noises and noise records from underwater noise fields.

1 - Introduction

Robust detection and estimation techniques have been developed for environments in which there is some uncertainty as to the exact distribution of the noise processes present[1,2,3]. In [4,5] the problem of detecting a signal in contaminated noise is discussed and the robust detector in terms of risk is found to be a censored probability likelihood test. The robust location estimator and small signal detector for nominal noise distributions having lighter than exponential tails (corresponding to strongly unimodal densities) are described in [3] and [6], respectively.

Huber in [3] developed the basic theory for robust estimation of location for convex classes of distribution functions. The motivation for this work was the the desire to find a technique which diminished the inordinate influence of outliers on estimates. He found that this could be done by using the system which would give the MLE for location if the noise had the distribution with the smallest Fisher's Information in the given convex class of distributions. The discussion in §**2** lifts his restriction on the distributions; it contains an investigation of the analogous robust detector and location estimator for the case where the nominal has heavier than exponential tails. The paper concludes with the results of some simulations which were performed with the robust detector developed in §**2**.

We will discuss robust detection of a small signal in independent identically distributed mixture noise. The distribution $\mu(\mathbf{x})$ of the noise is given by

$$\mu(\mathbf{x}) = \prod_{i=1}^{n} F(x_i) = \prod_{i=1}^{n} [(1-\epsilon)G(x_i) + \epsilon H(x_i)], \qquad 0<\epsilon<1 \tag{1.1}$$

where G is the known nominal distribution, H is the unknown contaminating distribution and F is the actual distribution of the individual observations. The univariate mixture model is given by

$$F = (1-\epsilon)G + \epsilon H \quad , \quad 0<\epsilon<1 \tag{1.2}$$

where G is the known nominal distribution and H is the unknown contaminant. Furthermore, if G and H have densities g and h, respectively, we can write

$$f = (1-\epsilon)g + \epsilon h \quad , \quad 0<\epsilon<1 \tag{1.3}$$

where f is the density of F.

2 - Robust Estimation and Detection for Noise with a Non-Strongly Unimodal Nominal Density

In [3,4] the least favorable distribution was developed for the mixture class of distributions which have a strongly unimodal nominal density. This type of density is limited to those which have lighter than exponential tails. However, many of the noise environments commonly encountered in atmospheric and oceanographic applications have been found [7,9,10] to have distributions which are much heavier tailed than exponential. It therefore seems proper to investigate the robust problem for very heavy tailed distributions.

We will find the robust small-signal detector and M-estimator for the mixture class of distributions having what will be called non-strongly unimodal nominal densities. This class is defined as follows.

Definition 1. *A unimodal density g is called non-strongly unimodal if it satisfies the following conditions:*

i) $sgn\left[\frac{-g'}{g}(x-c)\right] = sgn\,[x-c]$ *some real* c.

ii) There exist P_1 *and* P_2 *with* $P_2 < c < P_1$ *such that*

  a) For $x < P_2$ *and* $x > P_1$*, then* $-g'(x)/g(x)$ *is monotonically decreasing.*

  b) For $P_2 \leq x \leq P_1$*, then* $-g'(x)/g(x)$ *is monotonically increasing.*

iii) $\lim_{|x|\to\infty} g'(x)/g(x) = 0.$

For simplicity we will assume throughout this section that $c=0$. Note that if g is symmetric and $c=0$ then $P_2 = -P_1$.

Cauchy noise is an example of a distribution having a non-strongly unimodal density. Unfortunately, the moments of the Cauchy distribution are undefined. However, during many practical applications, the extreme tails would be truncated by the signal processor so that we can, if it seems appropriate, model a noise process as Cauchy within some finite interval of the observation space. Over such an interval all moments exist and are finite.

The members of the Johnson family [7,9,11] of distributions also have non-strongly unimodal distributions. Willett [9] found that a member of this class closely modeled the noise produced by shrimp along the ocean floor. Johnson noise is generated by the following memoryless transformation on Gaussian noise:

$$x = \lambda \sinh\left(\frac{z}{\delta}\right)$$

Here z is an observation from a unit normal process and x is an observation from the generated Johnson process. The parameter δ controls the tail weight; very large δ produces a nearly Gaussian process; and λ is a scale parameter given by

$$\lambda = \left[\frac{2\sigma^2}{e^{\frac{2}{\delta^2}} - 1}\right]^{1/2}$$

where σ^2 is the variance of the Johnson process. Because δ, and hence the tail mass, is allowed to vary continuously, the Johnson family can afford a good basis for approximating distributions with very heavy tails.

We will now investigate the properties of the absolutely continuous mixture density function f

$$f = (1-\epsilon)g + \epsilon h \tag{2.1}$$

where g is a non-strongly unimodal density and h is any density function for which the resulting f is absolutely continuous. We will also assume in this section that f is finite, unimodal and attains its maximum at $x=0$. This assumption is reasonable since many distributions encountered in nature are unimodal and we place the mode at zero without loss of generality. Hence, we have added the constraint that $\epsilon h'(x)$ is less(greater) than $-(1-\epsilon)g'(x)$ for

x greater(less) than zero.

Fact 1. *The density f is not strongly unimodal.*

Proof. From the definition of non-strongly unimodal we have

$$\lim_{|x|\to\infty} \frac{-g'}{g} = 0.$$

Then for any $\delta>0$ there exists finite $M>0$ such that $-g'(x)/g(x) < \delta$ for all $x>M$. Pick $z_0>M$. Since both f and g are finite we have $-\log f(z_0) + \log g(z_0) = r<\infty$. Furthermore, by the positivity of h, $-\log f(x) < -\log g(x)+c$, for all x, where $c= -\log(1-\epsilon)$. Assume f is strongly unimodal; that is, $-\log f$ is convex and, for positive x, dominates some line with constant positive slope k. Pick any δ between zero and k. Then for $z_1>z_0$

$$-\log f(z_1) + \log f(z_0) \geq k(z_1-z_0) \tag{2.2}$$

and

$$-\log g(z_1) + \log g(z_0) < \delta(z_1-z_0). \tag{2.3}$$

Combining (2.2) and (2.3) yields

$$\begin{aligned} -\log f(z_1) + \log g(z_1) &> -\log f(z_0) + \log g(z_0) + (k-\delta)(z_1-z_0) \\ &= r + (k-\delta)(z_1-z_0). \end{aligned} \tag{2.4}$$

But the right side of (2.4) can be made as large as desired by choosing z_1; hence it can be made larger than c and the positivity condition for h will be violated which, by contradiction, completes the proof.

Fact 2. *If $-f'/f$ has a limit*† *then*

$$\lim_{|x|\to\infty} \frac{-f'}{f} = 0 \tag{2.5}$$

The proof of Fact 2 is virtually identical to that of Fact 1.

In light of Facts 1 and 2 we see that we cannot contaminate a non-strongly unimodal density and produce one which is strongly unimodal. Furthermore, if f is not strongly unimodal then the derivative of $-f'/f$ must be negative for some positive arguments. In addition, since f is unimodal, $-f'(x)/f(x)$ is zero at the origin and positive for all positive values of x; therefore its derivative cannot be negative for all positive x. As a result it is not possible to fulfill the conditions of the lemma of the previous section if the nominal density of f is non-strongly unimodal.

Fact 3. *If the nonlinearity $\eta(x)$ is zero at $x=\pm\infty$ and at $x=0$ and if $sgn[\eta(x)] = sgn[x]$ then $E_F\eta' >0$.*

Proof. Since $\eta(x)>0$ for $x>0$ and $\eta(x)<0$ for $x<0$, then $\eta'(x)$ must be positive for some interval (b,a) about the origin. Furthermore,

$$\int_{-\infty}^{\infty} \eta' = 0$$

Then, after separating the real line into $(-\infty,b]$, (b,a) and $[a,\infty)$ we have

$$0 < \int_b^a \eta' = -\int_{-\infty}^{b} \eta' - \int_a^{\infty} \eta'. \tag{2.6}$$

By the unimodality of f we have

$$\int_{-\infty}^{\infty} \eta' f > f(b)\Big[\int_b^0 \eta' + \int_{-\infty}^{b} \eta'\Big] + f(a)\Big[\int_0^a \eta' + \int_a^{\infty} \eta'\Big] = 0 \tag{2.7}$$

---

† For an example of a mixture density f for which $-f'/f$ has no limit see [12].

which completes the proof.

Let us now apply Fact 3. We are given two distribution functions F_1 and F_2 both of the form

$$F_i = (1-\epsilon)G + \epsilon H_i, \qquad i=1,2.$$

with density functions $f subi(x) = dF_i(x)/dx$. Let $A = \{x\colon \eta'(x)<0\}$ and suppose

$$\int_A h_1 = 1 \text{ and } \int_A h_2 = 0.$$

Then

$$E_{F_1}\eta' < E_{F_2}\eta'.$$

where as a consequence of Fact 3

$$|E_{F_i}\eta'| = E_{F_i}\eta', \qquad i=1,2. \tag{2.8}$$

Hence the numerator of the expression for efficacy and the denominator of that for asymptotic variance (see [3,12]) are minimized by F_1. Similarly, if we let $B = \{x\colon |\eta(x)| = a\ maximum\}$ and suppose

$$\int_B h_1 = 1 \text{ and } \int_B h_2 = 0.$$

Then

$$E_{F_1}\eta^2 < E_{F_2}\eta^2. \tag{2.9}$$

Therefore, the denominator of the expression for efficacy and the numerator of that for asymptotic variance are maximized by F_1.

If $A = B$ then η must be decreasing on the set where it is at its maximum, a condition which can only be satisfied if the A and B are identical sets of discrete points. As a result, the contaminant H_1 would put all of its mass on a set of Lebesgue measure zero and would therefore be degenerate. Since we have constrained the problem so that a degenerate contaminant is not allowable we can state Fact 4 as follows.

Fact 4. *Let C be the set of all mixture distributions having finite, absolutely continuous, unimodal densities. If η satisfies the conditions given in Fact 3, then, for a non-strongly unimodal nominal density, it is not possible to maximize $E_F\eta^2$ and minimize $E_F\eta'$ with the same distribution $F \in C$.*

The underlying problem that Fact 4 formalizes is that we have no clear cut method by which to decide whether to put the contaminant mass where the nonlinearity is large in magnitude or where its derivative is negative. We will, in the remainder of this section, investigate the robust problem under the constraint that the support of the contaminant is limited to an interval. By limiting the allowable class of contaminants in this fashion we will be able to find a nonlinearity η_0 and a distribution $F_0 = (1-\epsilon)G + \epsilon H_0$ which satisfy

$$\min_{F \in C_*} E_F \eta_0{}' = E_{F_0}\eta_0, \tag{2.10}$$

$$\max_{F \in C_*} E_f \eta_0^2 = E_{F_0}\eta_0^2 \tag{2.11}$$

and

$$\eta_0 = \frac{-f_0'}{f_0}$$

where C_* contains all mixtures with contaminants whose support is confined within some specified finite interval. Clearly, if Eqs. (2.10) and (2.11) are satisfied, then

$$\min_{F \in C_*} I(F) = I(F_0),$$

that is, F_0 is the minimum Fisher's Information distribution in C_*.

The robust detector and estimator for the class C_*

Let g be a non-strongly unimodal density function as in Definition 1 with $c=0$. Define the function

$$p(x) = \begin{cases} g(a_1)e^{k(x-a_1)}, & a_1 \leq x < b_1 \\ g(a_2)e^{k(a_2-x)}, & a_2 \geq x > b_2 \\ g(x), & otherwise \end{cases} \tag{2.12}$$

where

$$k = \frac{-g'}{g}(a_1) = \frac{g'}{g}(a_2) \tag{2.13}$$

and $P_2 < a_2 < a_1 < P_1$ (see Definition 1).

Fact 5.

$$\left|\frac{-g'}{g}(a_i)\right| > \left|\frac{-g'}{g}(b_i)\right|. \tag{2.14}$$

Proof. We prove the fact for $i=1$; the proof for $i=2$ is virtually identical. From the definition of p, we have $p(a_1)=g(a_1)$ and $p(b_1)=g(b_1)$; hence $-log\ p(a_1) = -log\ g(a_1)$ and $-log\ p(b_1) = -log\ g(b_1)$. Therefore,

$$\int_{a_1}^{b_1} [\frac{-g'}{g}(x) - \frac{-g'}{g}(a_1)]\ dx = 0. \tag{2.15}$$

By the definition of g the integrand on the left side of (2.4.15) is positive for $a_1 < x < P_1$. Therefore, the integrand must be negative for some x between P_1 and b_1. But

$$\min_{x \in (P_1, b_1)} \frac{-g'}{g}(x) = \frac{-g'}{g}(b_1)$$

so the integrand must be negative at b_1, completing the proof.

Note that

$$\frac{-p'}{p}(x) = \begin{cases} k, & a_1 \leq x < b_1 \\ -k, & b_2 < x < a_2 \\ \frac{-g'}{g}(x), & otherwise \end{cases} \tag{2.16}$$

which is non-decreasing on $S_1 = (b_2, b_1)$, non-increasing on $S_2 = (-\infty, b_2] \cup [b_1, \infty)$ and is constant at its maximum magnitude on $A = (b_2, a_2] \cup [a_1, b_1)$.

Fact 6 *Given two functions p_α and p_ω defined as in (2.15). If*
(1) $k_\alpha < k_\omega$ then
(2) $|a_{i_\alpha}| < |a_{i_\omega}|$ for i=1,2.
(3) $-log\ p_\alpha(x) \leq -log\ p_\omega(x)$ for all x.
(4) $|b_{i_\alpha}| > |b_{i_\omega}|$ for i=1,2.

Proof. By Definition 1 and Eqs. (2.12) and (2.13)
(1) => (2),
(1) and (2) => (3),
and (3) => (4), completing the proof.

Assume, without loss of generality, that

$$\frac{-g'}{g}(P_1) \geq \frac{-g'}{g}(P_2)$$

Let a_2 be chosen as the smallest number in (P_2,c) such that

$$\frac{g'}{g}(a_2) \leq \frac{-g'}{g}(a_1) \tag{2.17}$$

Define the function

$$R(a_1) = \int_A [p - g] \quad 0 < a_1 < P_1 \tag{2.18}$$

where $A = (b_2,a_2] \cup [a_1,b_1)$.

Lemma. *If g and $-g'/g$ are continuous, then $R(a_1)$ is a monotonically non-increasing continuous function of a_1 with range $(0,\infty)$.*

Proof. See [12]

The main result of this section is given by the following theorem.

Theorem. *The minimum Fisher Information mixture distribution having the non-strongly unimodal nominal density g and contaminant supported on (b_2,b_1) has the density*

$$f_0 = (1-\epsilon)g + \epsilon h_0. \tag{2.19}$$

The density of of the contaminant is given by

$$\epsilon h_0(x) = \begin{cases} (1-\epsilon)[g(a_1)e^{k(a_1-x)} - g(x)] & a_1 \leq x < b_1 \\ (1-\epsilon)[g(a_2)e^{k(x-a_2)} - g(x)] & b_2 < x \leq a_2 \\ 0 & \textit{otherwise} \end{cases} \tag{2.20}$$

where

$$k = \frac{(1-\epsilon)\left\{g(a_1)[1-e^{k(a_1-b_1)}] + g(a_2)[1-e^{k(b_2-a_2)}]\right\}}{\epsilon + (1-\epsilon)\left\{\int_{a_1}^{b_1} g\ dx + \int_{b_2}^{a_2} g\ dx\right\}} \tag{2.21}$$

$$= \frac{-g'}{g}(a_1) = \frac{g'}{g}(a_2)\ ,$$

and the constants b_i, $i=1,2$ are found from

$$\log g(a_i) + \frac{g'}{g}(a_i)\,|x - a_i| - \log g(b_i) = 0\ ; \tag{2.22}$$

$b_2 < P_2 < a_2 < a_1 < P_1 < b_1$. If g and $-g'/g$ are continuous then a minimum Fisher's Information distribution in the form of Eqs. (2.18)-(2.20) exists for all ϵ between 0 and 1.

Proof. See [12]

3 - Numerical Results

In §2 we found the robust detector and estimator for the class of mixture noise distributions having a non-strongly unimodal nominal density g. Because the function $-g'/g$ was not monotonic we found it necessary to limit the support of the contaminant to an interval about the mode of the nominal density. For distributions satisfying this constraint the worst distribution in terms of Fisher's Information is given by Eqs. (2.19)-(2.21). The question which immediately comes to mind is: how does the robust detector perform if we violate the constraint on the support of the contaminant?

For the duration of this section we will confine our investigation to the robust detector; the analagous conclusions which can be drawn for the estimation problem should be apparent. In this section we will introduce some numerical results which will lend some insight into the question posed at the end of the preceding paragraph. We will look at three different but related situations. In the first, we will investigate the performance of several different detectors when the mixture noise conforms to the restriction on the support of the contaminant. Secondly, we relax this restriction so that the interval on which the contaminant is supported is a superset of the interval which was previously allowed. We then look at the performance of the same detectors as in the first case. Lastly, we investigate the performance of the robust detector for several different distributions which have support over the entire real line.

All of the figures discussed in this section plot efficacy against the natural logarithm of the mixing constant ϵ. For simplicity, we assume symmetric noise densities. The values for efficacy were computed using numerical integration.

Figs. 1-4 plot the efficacies for the optimal, robust, linear and sign detectors. The nominal density of the noise for these calculated values is given by

$$g(x) = \frac{1}{\sqrt{2\pi}}\frac{\delta}{\lambda}[1+(\frac{x}{\lambda})^2]^{-1/2} \exp\{-1/2[\delta \sinh^{-1}(\frac{x}{\lambda})]^2\} \tag{3.1}$$

which is the expression for a Johnson density with parameter δ (see the discussion of Johnson densities at the beginning of §2). For this work we used $\delta=1$ and $\sigma=1$. The contaminant density is given by

$$h(x) = \begin{cases} \dfrac{\frac{1}{\sqrt{2\pi}\sigma_G}[e^{-\frac{x^2}{2\sigma_G{}^2}} - e^{-\frac{b^2}{2\sigma_G{}^2}}]}{\int_{-b}^{b}\frac{1}{\sqrt{2\pi}\sigma_G}\left[e^{-\frac{z^2}{2\sigma_G{}^2}} - e^{-\frac{b^2}{2\sigma_G{}^2}}\right]dz}, & -b < x < b \\ 0, & otherwise \end{cases} \tag{3.2}$$

where, due to the symmetry we have imposed, b_2 and b_1 from (2.20) have become $-b$ and b, respectively. For Figs. 1 and 3 $\sigma_G=1$ and for Figs. 2 and 4 $\sigma_G=10$.

As we know from §2, the efficacy of the robust detector cannot be any lower for any member of the class of mixture distributions satisfying the constraint on the support of the contaminant than it is for the worst distribution given by Eqs. (2.19)-(2.21). However, this tells us nothing about how *well* the detector can perform. Figs. 1 and 2 show that for the mixture density given by (3.1) and (3.2) the performance is not appreciably worse for the robust detector than it is for the detector which is optimal for those densities. Note that in Fig. 2 the performance is slightly worse than it is in Fig. 1. This appears to be due to the fact that for higher σ_G more of the contaminant mass falls within the intervals $(-b, -a]$ and $[a, b)$. Within $(-b, b)$, the nonlinearity for the robust detector has minimum slope and maximum magnitude on these two intervals. From the discussion in the previous section, it is clear that this will tend to reduce the efficacy of the detector when compared with the case wherein the contaminant mass is more concentrated around the mode of the nominal density.

Figs. 3 and 4 plot the efficacy of the robust detector for the noise given by Eqs. (3.1) and (3.2) along with that for the sign and linear detectors. From these graphs we can readily see that the performance of the robust detector is far superior to that of the others, especially that of the linear detector. Both the linear detector and the sign detector have been widely used and, at least in this case, it is apparent that better results are readily obtainable.

Allowing the support of the contaminant to exceed the the interval $(-b, b)$ by a small amount appears to have little effect on performance. In Figs. 5-8 the contaminant is supported on $(-b - 1/10, b + 1/10)$; that is, it exceeds $|b|$ by ten percent of the standard deviation of the nominal, which in this case is 0.1. There is a slight but largely insignificant change in the performance from the situation where the contaminant is confined to $(-b, b)$.

Figs. 9 and 10 compare efficacy for the robust detector in two different mixture noise environments. The first noise has the worst distribution as described in §**2**; the second has the same non-strongly unimodal nominal as the worst density and a contaminant which is supported on the entire real line. If, in fact, there exists a distribution which has minimum Fisher's Information for all mixtures with a given nominal then it follows that the detector which is optimal for that distribution must have its minimum efficacy for that same distribution; that is, it must satisfy the saddlepoint condition. This is the crux of the robust problem; to find a lower bound on the performance of a device for a set of possible circumstances under which the device may be operated. While we have not shown that the distribution given by the theorem of §**2** has minimum Fisher's Information for the set of *all* mixture distributions having a given non-strongly unimodal nominal (we have done so for those with contaminants having bounded support on (b_2, b_1)) we can still find a lower bound for the performance for distributions having specified contaminants.

In Figs. 9 and 10 we show the efficacy for the robust detector for the worst noise distribution and for a noise distribution with contaminant supported along the entire real line. In both cases the nominal distribution is Johnson with $\delta=1$ and $\sigma=1$. The contaminant is Normal in Fig. 9 and is Johnson($\delta=1$) in Fig. 10. In both cases the variance of the contaminant is unity. We have also investigated a number of other contaminants including Cauchy, Laplace, Logistic, and Johnson with various values of δ. The results are very similar to Figs. 9 and 10. From the figures it is apparent that, for the set of contaminants tested, the worst distribution provides a lower bound on performance, as we would expect.

ACKNOWLEDGMENT

This research was supported by the Office of Naval Research under Grant N00014-81-K-0146 and by the U.S. Army under Grant DAAG29-85-K-0191.

REFERENCES

[1] J. W. Tukey, A survey of sampling from contaminated distributions, in *Contributions to Probability and Statistics* (ed. Olkin), pp. 448-485, Stanford University Press, 1960.

[2] F. R. Hampel, "A general qualitative definition of robustness," *Ann. Math. Statist.*, vol. 42, no. 6, pp. 1887-1896, 1971.

[3] P. J. Huber, "Robust estimation of a location parameter," *Ann. Math. Statist.*, vol. 35, pp. 73-101, Mar. 1964.

[4] ----, "A robust version of the probability ratio test," *Ann. Math. Statist.*, vol. 36, pp. 1753-1758, Dec. 1965.

[5] R. D. Martin and S. C. Schwartz, "Robust detection of a known signal in nearly Gaussian noise," *IEEE Trans. Inform. Theory*, vol. IT-17, pp. 50-56, Jan. 1971.

[6] S. A. Kassam and J. B. Thomas, "Asymptotically robust detection of a known signal in contaminated non-Gaussian noise," *IEEE Trans. Inform Theory*, vol. IT-22, pp. 22-26, Jan. 1976.

[7] E. Modugno, "The detection of signals in impulsive noise," *Ph.D. Dissertation*, Dept. of Electrical Engineering and Computer Science, Princeton University, Princeton, NJ, 1982.

[8] H. L. Royden, *Real Analysis,* Macmillan Publishing Co., Inc., pp. 157-180, New York, 1968.

[9] P. K. Willett, "The analysis of some undersea noise with applications to detection," to appear in *Proceedings of the 24th Allerton Conference on Communication, Control and Computing,* Oct. 1986.

[10] J. Veitch, "A characterization of Arctic undersea noise," *J. Acoust. Soc. Am.*, vol. 77, pp. 989-999, March 1985.

[11] N. L. Johnson and S. Kotz, *Distributions in Statistics: Continuous Univariate Distributions,* John Wiley and Sons, New York, 1971.

[12] D. Warren, "Topics in robust estimation and detection and in quasi-optimal detection," *Ph.D. Dissertation,* Department of Electrical Engineering, Princeton University, Princeton, NJ, 1986.

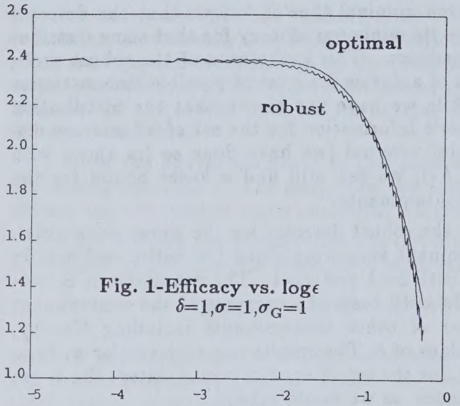


Fig. 1-Efficacy vs. logϵ
$\delta=1,\sigma=1,\sigma_G=1$

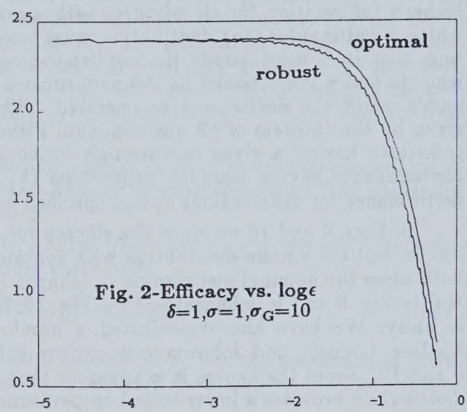


Fig. 2-Efficacy vs. logϵ
$\delta=1,\sigma=1,\sigma_G=10$

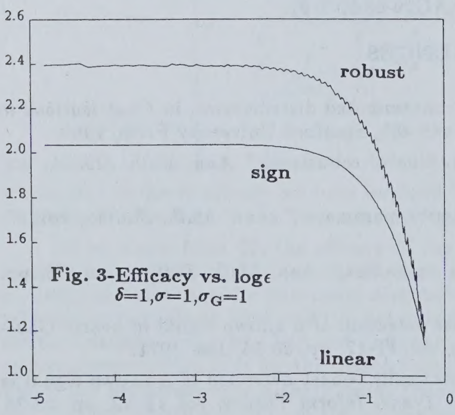


Fig. 3-Efficacy vs. logϵ
$\delta=1,\sigma=1,\sigma_G=1$

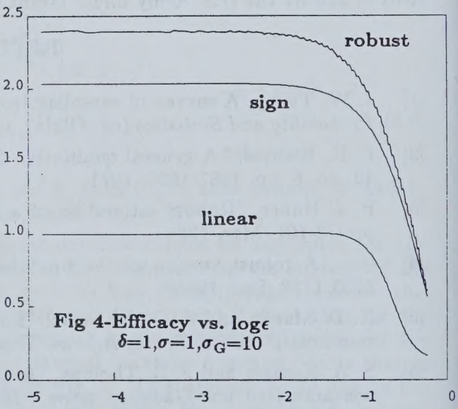


Fig 4-Efficacy vs. logϵ
$\delta=1,\sigma=1,\sigma_G=10$

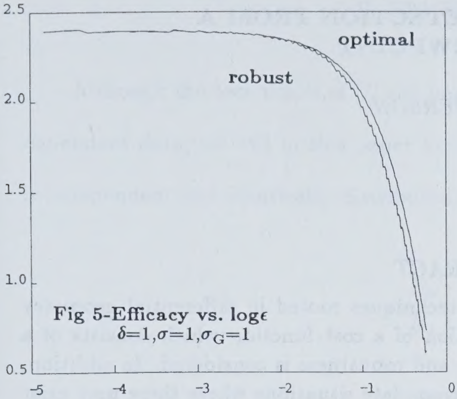


Fig 5-Efficacy vs. logϵ
δ=1,σ=1,σ_G=1

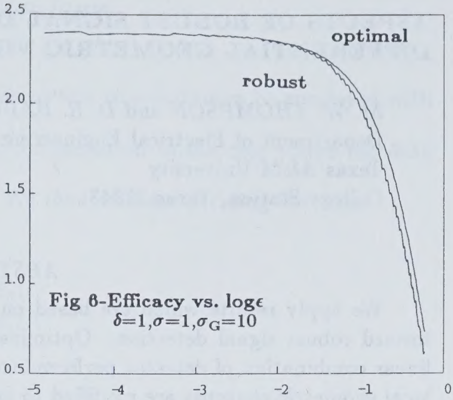


Fig 6-Efficacy vs. logϵ
δ=1,σ=1,σ_G=10

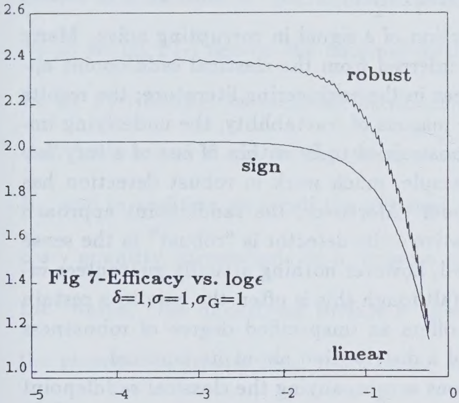


Fig 7-Efficacy vs. logϵ
δ=1,σ=1,σ_G=1

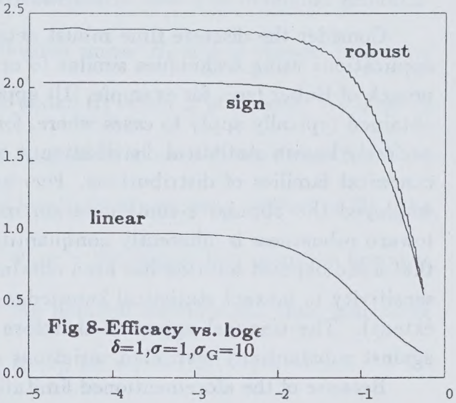


Fig 8-Efficacy vs. logϵ
δ=1,σ=1,σ_G=10

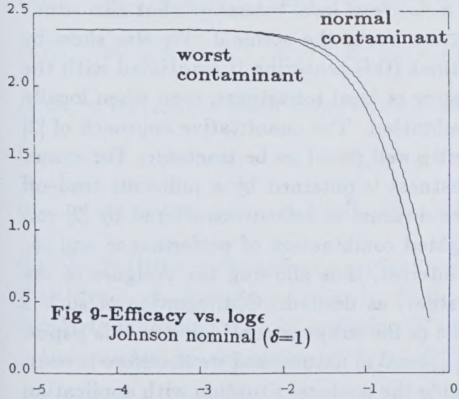


Fig 9-Efficacy vs. logϵ
Johnson nominal (δ=1)

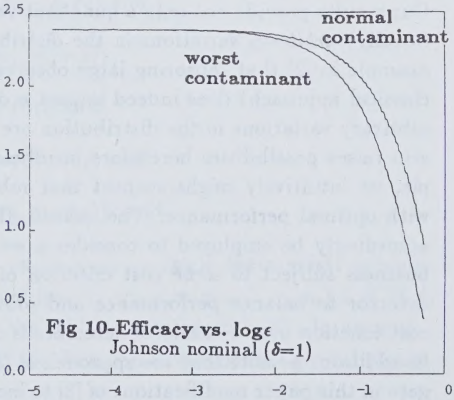


Fig 10-Efficacy vs. logϵ
Johnson nominal (δ=1)

ASPECTS OF ROBUST SIGNAL DETECTION FROM A DIFFERENTIAL GEOMETRIC VIEWPOINT

M. W. THOMPSON and D. R. HALVERSON
Department of Electrical Engineering
Texas A&M University
College Station, Texas 77843

ABSTRACT

We apply results which are based on techniques rooted in differential geometry toward robust signal detection. Optimization of a cost function which consists of a linear combination of detector performance and robustness is considered. In addition, local geometric concepts are modified to accomodate situations where there may exist greater uncertainty regarding the underlying noise distribution.

I. INTRODUCTION

Consider the discrete time robust detection of a signal in corrupting noise. Many applications using techniques similar to or inferred from the classical saddlepoint approach of Huber (see, for example, [1]) appear in the engineering literature; the results obtained typically apply to cases where, for reasons of tractablility, the underlying imperfectly known statistical distribution is constrained to lie within of one of a very few canonical families of distributions. For example, much work in robust detection has employed the popular ε-contamination model. Moreover, the saddlepoint approach toward robustness is inherently nonquantitative – the detector is "robust" in the sense that a saddlepoint solution has been obtained, however nothing actually guarantees insensitivity to inexact statistical knowledge (although this is often the case to a certain extent). The classical approach therefore offers an unspecified degree of robustness against substantially restricted variations of a distribution about its nominal.

Because of the aforementioned limitations accompanying the classical saddlepoint approach toward imparting robustness, we have considered in [2] an alternative approach which views the subject of robustness from a differential geometric prospective. Our results provide not only a quantitative measure of local robustness but also admit (locally) *arbitrary* variations in the distribution about the nominal. We also show by example in [2] that censoring large observations (this censoring is associated with the classical approach) does indeed impart a degree of local robustness, even when locally arbitrary variations in the distribution are admitted. The quantitative approach of [2] also raises possibilities heretofore insufficiently well posed to be tractable. For example, we intuitively might suspect that robustness is obtained by a judicious tradeoff with optimal performance. The quantitative measure of robustness offered by [2] can accordingly be employed to consider a weighted combination of performance and robustness subject to some cost criterion of interest, thus allowing the designer of the detector to balance performance and robustness as desired. Optimization of such a cost function over a family of detectors is one of the subjects considered in this paper. In addition, we note that the approach of [2] is local in nature, and we therefore investigate in this paper modifications of [2] to include the nonlocal situation with application to cases where substantial uncertainty is present in the underlying distribution.

II. DEVELOPMENT

Although the techniques of [2] can under certain circumstances be employed with dependent data, we will in this paper limit consideration to the case where the data is independent and identically distributed. We thus consider deciding between

$$\begin{array}{l} H_0 : F_0(\cdot) \\ H_1 : F_1(\cdot) \end{array},$$

where $F_i(\cdot)$ is the univariate distribution of n i.i.d samples under $H_i, i = 0, 1$. We choose as a measure of "performance" either false alarm rate α or detection probability β; we let $F(\cdot)$ denote the nominal distribution under H_0 if α is chosen, otherwise we let $F(\cdot)$ denote the nominal distribution under H_1 when β is chosen. We observe that both α and β can be expressed as an integral over an n-dimensional Borel set $\mathcal{B}_n$, and in addition we recall the measure of local robustness $\cos\gamma$ defined in [2]. The $\cos\gamma$ quantity corresponds to a measure of slope – the bigger this quantity becomes the "flatter" the underlying surface is near the nominal distribution, thus indicating the presence of greater local robustness. As in [2], for $i, j = 1, 2, \ldots, n$, we let

$\partial_i^+ \mathcal{B}_n = \{(x_1, x_2, \ldots, x_n)$: there exists $\varepsilon > 0$ such that

$(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_n) \in \overline{\mathcal{B}}_n$ for $y \in (x_i - \varepsilon, x_i)$ and

$(x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_n) \in \mathcal{B}_n$ for $z \in (x_i, x_i + \varepsilon)\}$,

$\partial_i^- \mathcal{B}_n = \{(x_1, x_2, \ldots, x_n)$: there exists $\varepsilon > 0$ such that

$(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_n) \in \mathcal{B}_n$ for $y \in (x_i - \varepsilon, x_i)$ and

$(x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_n) \in \overline{\mathcal{B}}_n$ for $z \in (x_i, x_i + \varepsilon)\}$, and

$\partial_{ij}^{++} \mathcal{B}_n = \{(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n, y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_n)$: there exists w such that $(x_1, \ldots, x_{i-1}, w, x_{i+1}, \ldots, x_n) \in \partial_i^+ \mathcal{B}_n$ and $(y_1, \ldots, y_{j-1}, w, y_{j+1}, \ldots, y_n) \in \partial_j^+ \mathcal{B}_n\}$. Similarly, we define ∂_{ij}^{+-} by replacing ∂_j^+ in the ∂_{ij}^{++} expression with ∂_j^-. In an analogous manner, we also define ∂_{ij}^{-+} and ∂_{ij}^{--}. We then recall the following result from [2] which gives a closed form expression for $\cos\gamma$.

Theorem 1: Suppose that if $x \in \mathcal{R}^n \cap \overline{\partial_1^+ \mathcal{B}_n} \cap \overline{\partial_2^+ \mathcal{B}_n} \cap \ldots \cap \overline{\partial_n^+ \mathcal{B}_n} \cap \overline{\partial_1^- \mathcal{B}_n} \cap \overline{\partial_2^- \mathcal{B}_n} \cap \ldots \cap \overline{\partial_n^- \mathcal{B}_n}$ then $x \in \text{int}(\mathcal{B}_2) \cup \text{int}(\overline{\mathcal{B}}_n)$. We then have $\cos\gamma = (1 + \Delta^2 + \Gamma)^{-1/2}$, where

$$\begin{aligned}
\Delta^2 = \sum_{i=1}^{n} \Bigg[& \int_{\partial_i^+ \mathcal{B}_n} dF(y_1)\ldots dF(y_{i-1})dF(y_{i+1})\ldots dF(y_n) \\
& - \int_{\partial_i^- \mathcal{B}_n} dF(y_1)\ldots dF(y_{i-1})dF(y_{i+1})\ldots dF(y_n) \Bigg]^2 \\
\Gamma = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \Bigg[& \int_{\partial_{ij}^{++} \mathcal{B}_n} dF(x_1)\ldots dF(x_{i-1})dF(x_{i+1})\ldots dF(x_n) \\
& \qquad \cdot dF(y_1)\ldots dF(y_{j-1})dF(y_{j+1})\ldots dF(y_n) \\
& + \int_{\partial_{ij}^{--} \mathcal{B}_n} dF(x_1)\ldots dF(x_{i-1})dF(x_{i+1})\ldots dF(x_n) \\
& \qquad \cdot dF(y_1)\ldots dF(y_{j-1})dF(y_{j+1})\ldots dF(y_n) \\
& - \int_{\partial_{ij}^{-+} \mathcal{B}_n} dF(x_1)\ldots dF(x_{i-1})dF(x_{i+1})\ldots dF(x_n) \\
& \qquad \cdot dF(y_1)\ldots dF(y_{j-1})dF(y_{j+1})\ldots dF(y_n) \\
& - \int_{\partial_{ij}^{+-} \mathcal{B}_n} dF(x_1)\ldots dF(x_{i-1})dF(x_{i+1})\ldots dF(x_n) \\
& \qquad \cdot dF(y_1)\ldots dF(y_{j-1})dF(y_{j+1})\ldots dF(y_n) \Bigg],
\end{aligned}$$

whenever the integrals exist.

Proof: The above follows from the Theorem of [2].

QED

As alluded to in the Introduction, the measure of robustness provided by $\cos\gamma$ makes feasible the tradeoff between performance (as measured by α or β) and robustness subject to a cost criterion of interest. In this paper we illustrate the application of this approach by considering the case of the detection of a constant signal s in i.i.d. zero mean unit variance Gaussian noise. In order to impart robustness, we employ a family of detectors which modify the Neyman-Pearson optimal detector by admitting censoring, i.e. the detector nonlinearity $g(\cdot)$ is given by, for each $k > 0$,

$$g(x) = \begin{cases} k & \text{if } x > k \\ x & \text{if } |x| \leq k \\ -k & \text{if } x < -k \end{cases}, \qquad \text{for all } x \in \mathcal{R} .$$

For a prescribed false alarm rate α, we choose, for each k, the detector threshold T to achieve α when the underlying distribution is nominal under H_0. Allowing k to vary over R^+, we then choose the value of k (and thus the corresponding detector) which optimizes the linear cost function

$$J_a(k) = a\beta + (1-a)\cos\gamma \qquad \text{for } a \in [0,1).$$

When $n = 2$ the calculations are reasonably compact and we obtain
$\mathcal{B}_2 = \{(x,y) : x = T - k, y > k;\ \text{or } y = T - x, T - k \leq x \leq k;\ \text{or } y = T - k, x > k\}$
$\partial_1^+ \mathcal{B}_2 = \{(x,y) : x = T - k, y > k;\ \text{or } y = T - x, T - k \leq x < k\}$
$\partial_2^+ \mathcal{B}_2 = \{(x,y) : y = T - k, x > k;\ \text{or } y = T - x, T - k < x \leq k\}$
$\partial_1^- \mathcal{B}_2 = \partial_2^- \mathcal{B}_2 = \phi$
$\partial_{12}^{++} \mathcal{B}_2 = \partial_{21}^{++} \mathcal{B}_2 = \{(x,y) : x \geq k \text{ and } y \geq k,\ \text{or } y = x \text{ and } T - k < x < k\}$
$\partial_{12}^{+-} \mathcal{B}_2 = \partial_{21}^{+-} \mathcal{B}_2 = \partial_{12}^{-+} \mathcal{B}_2 = \partial_{21}^{-+} \mathcal{B}_2 = \partial_{12}^{--} \mathcal{B}_2 = \partial_{21}^{--} \mathcal{B}_2 = \phi.$
Noting that $\beta = \int_{\mathcal{B}_2} \frac{1}{2\pi} e^{-(x^2+y^2)/2} dx dy$ and that $\cos\gamma$ is given by Theorem 1, optimization over k of $J_a(k)$ can then be achieved for various values of a. The results are summarized for $\alpha = 0.05$ with $s = 0.1$ and $s = 0.5$ in Table 1 and Table 2 respectively.

Table 1

Optimal values, $s = 0.1$

a	k	$a\beta/[(1-a)\cos\gamma]$	$J_a(k)$
0	1.206	0	0.917
0.3	1.208	0.031	0.917
0.5	1.211	0.071	0.661
0.7	1.219	0.166	0.321
0.9	1.265	0.641	0.151
0.95	1.314	1.360	0.108
0.975	1.427	2.793	0.087

Table 2

Optimal values, $s = 0.5$

a	k	$a\beta/[(1-a)\cos\gamma]$	$J_a(k)$
0	1.286	0	0.824
0.3	1.306	0.085	0.626
0.5	1.333	0.199	0.494
0.7	1.382	0.692	0.362
0.9	1.692	1.923	0.233

Note that, somewhat surprisingly, maximum robustness ($a = 0$) is not attained by maximal censoring ($k \to 0$). This is in part a consequence of the interplay between k and T (for fixed T it is not difficult to show $k \to 0$). In addition, when the ratio $a\beta/[(1-a)\cos\gamma] = 1$ we have essentially equal weighting of performance and robustness. For the example above when $s = 0.1$ this occurs when a is between 0.9 and 0.95, and corresponds to a censoring value of k between 1.2648 and 1.3144. When $s = 0.5$ in the example, this occurs when a is between 0.7 and 0.9, and corresponds to a censoring value of k between 1.382 and 1.692. Thus when $s = 0.1$ the amount of censoring is near that corresponding to maximum robustness unless performance is disproportionately weighted, although with the stronger signal this is not so much the case and less censoring (larger k) is required if performance is given at least as much weighting as robustness.

III. NONLOCAL ROBUSTNESS

As discussed previously, it is desirable to extend the measure of robustness described in [4] to situations where there exists greater uncertainty regarding the underlying noise distribution. Consider first the case where the class of n-dimensional distribution functions is parameterized by $m \in \mathcal{N}$ parameters, $a_1, a_2, \ldots a_m$. From [2], we have that a local measure of robustness is given by (taking h as α or β)

$$\cos\gamma_m = (1 + \sum_{i=1}^{m} (\frac{\partial h}{\partial a_i})^2)^{-1/2},$$

where the associated derivatives are evaluated at the point corresponding to the nominal distribution. To account for a greater degree of uncertainty regarding the parameters, we will allow the parameters to vary over some specified region. Hence, a natural extension of the previous expression leads to the robustness measure defined by (see [3] for a description of terminology)

$$\Phi_m = \frac{\int_{R_m} (cos\gamma_m)^{-2}\Omega}{\int_{R_m} \Omega} = \frac{\int_{R_m} [1 + \sum_{j=1}^{m} (\frac{\partial h}{\partial a_j})^2]\Omega}{\int_{R_m} \Omega},$$

where R_m is a subset of $\mathcal{R}^m$ contained in an orientable p-dimensional submanifold $\mathcal{M}$ of $\mathcal{R}^m$ and parametrized by some cubical p-chain; Ω is a p-form which is the ordinary volume element of $\mathcal{M}$.

In this manner, the cumulative effects of the parameter variations within the region R_m (normalizing out the volume of R_m) are manifested in the measure of robustness. We note that in this case large values of Φ_m correspond to large degrees of nonrobustness, whereas small values of Φ_m correspond to large degrees of robustness.

As an example of the application of the aforementioned measure of nonlocal robustness, recall the example of [2] wherein the discrete time detection of a constant signal s in i.i.d. Gaussian noise with mean μ and variance σ^2 is considered. Noting potential uncertainty in our knowledge of s, μ, and σ^2, we may view the distribution under H_1 as parameterized by these three quantities. In [2] the local robustness of the linear detector is compared to that of a censored linear detector. An extension of the analysis of [2] can be used to compare Φ_3 for the linear detector and the censored linear detector, the latter employing the nonlinearity $g(\cdot)$ of the previous section. For example, when $n = 2$ suppose that nominally $s = 0.1$, $\mu = 0$, and $\sigma^2 = 1$, with $R_3 = \{(s, \mu, \sigma^2) : 0.05 \leq s \leq 0.15, -.05 \leq \mu \leq .05, 0.5 \leq \sigma^2 \leq 1.5\}$. As before, we choose the thresholds of the two detectors to yield a false alarm rate $\alpha = 0.05$ when the s, μ, σ^2 values are nominal. We then can employ the closed form expressions from [2] for $\frac{\partial\beta}{\partial s}$, $\frac{\partial\beta}{\partial\mu}$, $\frac{\partial\beta}{\partial\sigma^2}$ for the linear detector, as well as numerical integration, to obtain through a lengthy analysis $\Phi_3 = 1.071$ for the linear detector, whereas $\Phi_3 = 1.155$ for the censored detector with $k = 1$. Note that this indicates that under the nonlocal measure of robustness the linear detector is slightly more robust than the censored detector. It is also of interest to note that for the nominal parameters stated above, $\cos\gamma = 0.964$ (recall $\cos\gamma$ is the local measure of robustness) for the linear detector and $\cos\gamma = 0.972$ for the censored detector with $k = 1$. Thus under the local measure of robustness the censored detector is indicated to be more robust than the linear detector.

A similar approach can be taken for larger n, however for very large n the n-fold numerical integration accompanying β for the censored detector complicates the situation, and there may be some attractiveness in applying a large sample Gaussian approximation of the test statistic as done in [2]. However, care would need to be taken as one would be integrating an approximation, and it may be profitable to investigate more efficient numerical schemes to directly calculate Φ_3 in such cases.

In the general nonparameterized situation we proceed analogously to [2]. Employing a partition $\mathcal{P}$ of the real line, we approximate a given univariate distribution function with a simple function taking on one of the values $a_0 = 0, a_1, a_2, \ldots, a_m, a_{m+1} = 1$, where the corresponding partitioning points of $\mathcal{P}$ are $x_0, x_1, \ldots, x_{m+1}$. We then could choose for each m a region of interest R_m contained in Euclidean m-space containing $a_1, a_2, \ldots, a_m$. As the norm of $\mathcal{P}$ tends to zero, m tends to infinity and we might be tempted to define the nonlocal measure of robustness via $\tilde{\Phi} = \lim_{|\mathcal{P}|\to 0} \Phi_m$. However, it turns out that in important cases $\tilde{\Phi}$ may not exist because we are somewhat redundantly considering too many points within a region of interest to obtain convergence. A more delicate analysis is therefore often desirable in the generalized case, as illustrated by the following example.

Consider the family of distributions $F(\cdot)$ satisfying, for all $x \in \mathcal{R}$, $G(x) \leq F(x) \leq H(x)$ for some fixed continuous distributions $G(\cdot)$ and $H(\cdot)$. We seek a reasonable way to approach this family through a sequence of regions in Euclidean space which yield a convergent measure of robustness. In order to keep the analysis reasonably brief we illustrate here how a result for $n = 2$ can be obtained, although the method could be applied for larger n.

Choose a partition $\tilde{\mathcal{P}}$ of $\mathcal{R}$ consisting of $(w_1, w_2, \ldots, w_q)$ and note that elements of $\tilde{\mathcal{P}} \times \tilde{\mathcal{P}}$ can be viewed as vertices which generate partitioning rectangles of $\mathcal{R}^2$. For $q = 1, 2, \ldots$ there exist collections $\{E_{qr}\}_{r=1}^{\rho_q}$ of these rectangles such that

$$h = \lim_{|\tilde{\mathcal{P}}|\to 0} \int_{\cup_{r=1}^{\rho_q} E_{qr}} dF(z_1)dF(z_2).$$

Consider a real number $\varepsilon > 0$ small enough so that $[w_i - \varepsilon, w_i + \varepsilon]$ is properly contained within the union of the partitioning intervals of $\tilde{\mathcal{P}}$ adjacent to w_i for all $i = 1, 2, \ldots, q$, and so that the $[w_i - \varepsilon, w_i + \varepsilon]$ intervals are disjoint. Recall the aforementioned partition $\mathcal{P}$ of $\mathcal{R}$ and the values $a_1, a_2, \ldots, a_m$ which correspond to the partitioning points $x_1, x_2, \ldots, x_m$ of $\mathcal{P}$. For each q, choose a set of q real numbers $b_{1q}, b_{2q}, \ldots, b_{qq}$. We then define the region $R_{m\varepsilon q} = \{(a_1, a_2, \ldots, a_m) : G(x_i) \leq a_i \leq H(x_i)$ for all $i \in \{1, 2, \ldots, m\}$ such that $x_i \notin \cup_{j=1}^{q} [w_j - \varepsilon, w_j + \varepsilon]$; otherwise, if $i \in \{1, 2, \ldots, m\}$ and $x_i \in [w_k - \varepsilon, w_k + \varepsilon]$ for some k, then $a_i = b_{kq}\}$. Note that b_{kq}, $k = 1, 2, \ldots, q$, could be taken to be the nominal value of the distribution function (if a nominal has been chosen) at w_k, however there is no requirement that this be done. Let $\Phi_{m\varepsilon q}$ be the value of the aforementioned Φ_m quantity with $\cup_{r=1}^{\rho_q} E_{qr}$ replacing $\mathcal{B}_n$ and $R_{m\varepsilon q}$ replacing R_m. We then define the measure of robustness

$$\Phi = \lim_{|\tilde{\mathcal{P}}|\to 0} \lim_{\varepsilon\to 0} \lim_{|\mathcal{P}|\to 0} \Phi_{m\varepsilon q},$$

whenever this quantity exists. We remark that we can define $\partial_1^{+} \cup_{r=1}^{\rho_q} E_{qr}$, $\partial_{12}^{++} \cup_{r=1}^{\rho_q} E_{qr}$, etc. by replacing $\mathcal{B}_n$ with $\cup_{r=1}^{\rho_q} E_{qr}$ in the earlier definition of $\partial_1^{+} \mathcal{B}_n$, $\partial_{12}^{++} \mathcal{B}_n$, etc. We also let $y_{1,+,\text{sup}} = \sup\{y :$ there exists x such that $(x, y) \in \partial_1^{+} \mathcal{B}_2\}$, $y_{1,-,\text{sup}} = \sup\{y :$ there exists x such that $(x, y) \in \partial_1^{-} \mathcal{B}_2\}$, $y_{2,+,\text{sup}} = \sup\{y :$ there exists x such that $(y, x) \in \partial_2^{+} \mathcal{B}_2\}$, $y_{2,-,\text{sup}} = \sup\{y :$ there exists x such that $(y, x) \in \partial_2^{-} \mathcal{B}_2\}$, and when $i, j = 0, 1$ with $i \neq j$, $y_{i,j,+,+,\text{sup}} = \sup\{y : (y, y) \in \partial_{ij}^{++} \mathcal{B}_2\}$, $y_{i,j,+,-,\text{sup}} =$

$\sup\{y : (y,y) \in \partial_{ij}^{+-}\mathcal{B}_2\}$, $y_{i,j,-,+,\mathrm{sup}} = \sup\{y : (y,y) \in \partial_{ij}^{-+}\mathcal{B}_2\}$, $y_{i,j,-,-,\mathrm{sup}} = \sup\{y : (y,y) \in \partial_{ij}^{--}\mathcal{B}_2\}$. Moreover, similar quantites are defined with "inf" replacing "sup" everywhere "sup" appears above. We remark that the suprema and infima considered here are allowed to take on extended real values. Similarly we define $y_{1,+,sup,q}$, $y_{1,2,+,+,sup,q}$, etc. by replacing $\mathcal{B}_2$ with $\cup_{r=1}^{\rho_q} E_{qr}$ in the preceding definitions. We will say that the Borel set $\mathcal{B}_2 \subset \mathcal{R}^2$ is **Φ-regular** if and only if

$$\lim_{|\tilde{\mathcal{P}}|\to 0} \int_{\partial_{(\cdot)}^{(\cdot)} \cup_{r=1}^{\rho_q} E_{qr}} (dG + dH)(x) = \int_{\partial_{(\cdot)}^{(\cdot)} \mathcal{B}_2} (dG + dH)(x),$$

$$\lim_{|\tilde{\mathcal{P}}|\to 0} \int_{\partial_{(\cdot)(\cdot)}^{(\cdot)(\cdot)} \cup_{r=1}^{\rho_q} E_{qr}} (dG + dH)(x) = \int_{\partial_{(\cdot)(\cdot)}^{(\cdot)(\cdot)} \mathcal{B}_2} (dG + dH)(x),$$

$$\lim_{|\tilde{\mathcal{P}}|\to 0} y_{(\cdot),(\cdot),(\cdot),q} = y_{(\cdot),(\cdot),(\cdot)} \text{ and}$$

$$\lim_{|\tilde{\mathcal{P}}|\to 0} y_{(\cdot),(\cdot),(\cdot),(\cdot),(\cdot),q} = y_{(\cdot),(\dot{\cdot}),(\cdot),(\cdot),(\cdot)}.$$

We then can establish the following result:
Theorem 2: Suppose that $\mathcal{B}_2$ is Φ–regular. We then have

$$\begin{aligned}
\Phi = 1 &+ \frac{1}{4}\left[\left(\int_{\partial_1^+ \mathcal{B}_2} - \int_{\partial_1^- \mathcal{B}_2}\right)(dG + dH)(x)\right]^2 + \frac{1}{4}\left[\left(\int_{\partial_2^+ \mathcal{B}_2} - \int_{\partial_2^- \mathcal{B}_2}\right)(dG + dH)(x)\right]^2 \\
&+ \frac{1}{2}\Bigg[\Bigg(\int_{\partial_{12}^{++} \mathcal{B}_2} + \int_{\partial_{12}^{--} \mathcal{B}_2} + \int_{\partial_{21}^{++} \mathcal{B}_2} + \int_{\partial_{21}^{--} \mathcal{B}_2} \\
&- \int_{\partial_{12}^{+-} \mathcal{B}_2} - \int_{\partial_{12}^{-+} \mathcal{B}_2} - \int_{\partial_{21}^{+-} \mathcal{B}_2} - \int_{\partial_{21}^{-+} \mathcal{B}_2}\Bigg)(dG + dH)(x)(dG + dH)(y)\Bigg] \\
&+ \frac{1}{12}\left[\left(\lim_{y\to y_{1,+,sup}^-} - \lim_{y\to y_{1,+,inf}^+} + \lim_{y\to y_{1,-,sup}^-} - \lim_{y\to y_{1,-,inf}^+}\right)(H(y) - G(y))^2\right] \\
&+ \frac{1}{12}\left[\left(\lim_{y\to y_{2,+,sup}^-} - \lim_{y\to y_{2,+,inf}^+} + \lim_{y\to y_{2,-,sup}^-} - \lim_{y\to y_{2,-,inf}^+}\right)(H(y) - G(y))^2\right] \\
&+ \frac{1}{6}\Bigg[\Bigg(\lim_{y\to y_{1,2,+,+,sup}^-} - \lim_{y\to y_{1,2,+,+,inf}^+} + \lim_{y\to y_{2,1,+,+,sup}^-} - \lim_{y\to y_{2,1,+,+,inf}^+} \\
&+ \lim_{y\to y_{1,2,-,-,sup}^-} - \lim_{y\to y_{1,2,-,-,inf}^+} + \lim_{y\to y_{2,1,-,-,sup}^-} - \lim_{y\to y_{2,1,-,-,inf}^+} \\
&- \lim_{y\to y_{1,2,+,-,sup}^-} + \lim_{y\to y_{1,2,+,-,inf}^+} - \lim_{y\to y_{2,1,+,-,sup}^-} + \lim_{y\to y_{2,1,+,-,inf}^+} \\
&- \lim_{y\to y_{1,2,-,+,sup}^-} + \lim_{y\to y_{1,2,-,+,inf}^+} - \lim_{y\to y_{2,1,-,+,sup}^-} + \lim_{y\to y_{2,1,-,+,inf}^+}\Bigg)(H(y) - G(y))^2\Bigg]
\end{aligned}$$

whenever these quantities exist.

Proof: Proceding in a manner analagous to [2] we note that the approximate measure of performance is given by $\tilde{h} = \int_{\cup_{r=1}^{\rho_q} E_{qr}} d\tilde{F}(y_1)d\tilde{F}(y_2)$, where $\tilde{F}(\cdot)$ denotes a simple function approximation to the distribution function $F(\cdot)$. Again using techniques similiar to [2] we obtain an expression for $\sum_{i=1}^{l}(\frac{\partial \tilde{h}}{\partial a_l})^2$. Expanding $(\frac{\partial \tilde{h}}{\partial a_l})^2$, we obtain for each l two squared terms and a cross product. Integrating one squared term over the defined region (one must carefully consider all of the cases which this entails), and performing lengthy algebraic manipulations, we obtain a term ξ_l of form

$$\xi_l = \prod_{k=1}^{m}{}^{\sharp}(H(x_k) - G(x_k)) \sum_{i=1}^{m-1}{}^{\sharp} \sum_{j=1}^{m-1}{}^{\sharp} \frac{1}{4}\Big\{(H(x_{i+1}) - H(x_i))(H(x_{j+1}) - H(x_j))$$

$$+2(H(x_{i+1}) - H(x_i))(G(x_{j+1}) - G(x_j)) + (G(x_{i+1}) - G(x_i))(G(x_{j+1}) - G(x_j))\Big\} T^l_{i,j}$$

$$+ \prod_{k=1}^{m}{}^{\sharp}(H(x_k) - G(x_k)) \sum_{i=2}^{m}{}^{\sharp} \frac{1}{12}\Big\{(H(x_{i+1}) - G(x_{i+1}))^2 T^l_{i,i} - 2(H(x_i) - G(x_i))^2 T^l_{i,i-1}$$

$$+ (H(x_i) - G(x_i))^2 T^l_{i,i}\Big\} + S \qquad ,$$

where $\sum_{i=p}^{q}{}^{\sharp}$ should be interpreted to mean $\displaystyle\sum_{\substack{\{i\in\{p,p+1,\ldots,q-1,q\}:\\ x_1 \notin \cup_{j=1}^{q}[w_j-\varepsilon,w_j+\varepsilon]\}}}$, and the symbol $\prod_{i=p}^{q}{}^{\sharp}$ should be interpreted in a similiar fashion. Also, $T^l_{i,j}$ is defined as $T^l_{i,j} = (I_{\mathcal{B}_2}(x_{l-1},x_i) - I_{\mathcal{B}_2}(x_l,x_i))(I_{\mathcal{B}_2}(x_{l-1},x_j) - I_{\mathcal{B}_2}(x_l,x_j))$ where $I_{\mathcal{B}_2}(\cdot,\cdot)$ is the indicator function of the set $\mathcal{B}_2$. Now after normalizing, letting $|\mathcal{P}| \to 0$ and $\varepsilon \to 0$, and invoking the continuity of $G(\cdot)$ and $H(\cdot)$, we see that $S \to 0$ and

$$\sum_{l=1}^{m} \xi_l \to \frac{1}{4}\left[\left(\int_{\partial_1^+ \cup_{r=1}^{\rho_q} E_{qr}} - \int_{\partial_1^- \cup_{r=1}^{\rho_q} E_{qr}}\right)(dG + dH)(x)\right]^2$$

$$+ \frac{1}{12}\left[\left(\lim_{y\to y^-_{1,+,\sup,q}} - \lim_{y\to y^+_{1,+,\inf,q}} + \lim_{y\to y^-_{1,-,\sup,q}} - \lim_{y\to y^+_{1,-,\inf,q}}\right)(H(y) - G(y))^2\right]$$

Next, letting $|\tilde{\mathcal{P}}| \to 0$ and invoking Φ–regularity we have the same result as shown above with the exception the the $\partial_1^{(\cdot)} \cup_{r=1}^{\rho_q} E_{qr}$ is replaced with $\partial_1^{(\cdot)}\mathcal{B}_2$ and $y^{(\cdot)}_{1,(\cdot),(\cdot),q}$ is replaced with $y^{(\cdot)}_{1,(\cdot),(\cdot)}$. We thus obtain those integral terms of the conclusion of the Theorem which are evaluated over $\partial_1^{(\cdot)}\mathcal{B}_2$, and those limit terms of the conclusion of the Theorem which are evaluated as $y \to y^{(\cdot)}_{1,(\cdot),(\cdot)}$. The remaining terms found in the conclusion of the Theorem are obtained in a similar manner by considering the other squared term and the cross product.

QED

Consider now the application of Theorem 2 to the linear detector and the aforementioned censored linear detector (censored for $|x| > k$). These detectors could have arisen via the presence of a Gaussian nominal, however the distributions of $G(\cdot)$ and $H(\cdot)$ need not necessarily envelope a Gaussian distribution. Recalling from [2] that for the linear detector $\partial_1^+\mathcal{B}_2 = \partial_2^+\mathcal{B}_2 = \{(x,y) : y = T - x\}$, $\partial_{12}^{++}\mathcal{B}_2 = \partial_{21}^{++}\mathcal{B}_2 = \{(x,y) : y = x\}$, with all others being empty, we obtain via Theorem 2, $\Phi = 1 + 1 + 1 + 0 + 0 + 0 + 0 = 3$ (linear), whereas recalling the aforementioned corresponding quantities for the censored detector we obtain via Theorem 2,

$\Phi = 1 + \frac{1}{2}(2 - H(T-k) - G(T-k))^2 - \frac{1}{6}(H(T-k) - G(T-k))^2 + (2 - H(k) - G(k))^2 + \frac{1}{3}[(H(k) - G(k))^2 - (H(T-k) - G(T-k))^2]$ (censored). Recall from [2] that for the linear detector $(\cos\gamma)^{-2} = 3$, regardless of the nominal distribution, so our result for the nonlocal situation is consistent with what we would expect for the linear detector. Noting that $T - k \leq k$, we then conclude that the censored detector is more robust than the linear one if $\frac{3}{2}[2 - H(T-k) - G(T-k)]^2 + \frac{1}{3} < 2$, i.e. if $H(T-k) + G(T-k) > 2 - \frac{\sqrt{10}}{3} \approx 0.95$. On the other hand, we conclude that the linear detector is more robust than the censored one if $\frac{3}{2}(2 - H(k) - G(k))^2 - \frac{1}{2} > 2$, i.e. if $H(k) + G(k) < 2 - \sqrt{\frac{5}{3}} \approx 0.71$.

Consider now the ramifications the above results have toward the classical robust detection of a positive constant signal in i.i.d. nominally Gaussian noise. The censoring value k is set under H_0 independently of s; we then desire robustness under H_1. But the above shows that for sufficiently large s and $n = 2$ the linear detector is actually more robust than the censored detector. The situation is exacerbated by the tendency of the designer to err on the side of overestimating the contamination (to protect α), thus often choosing a smaller value of k than most appropriate. If in reality the region determined by $G(\cdot)$ and $H(\cdot)$ is rather tight about the nominal then s need not be very large for the linear detector to possess greater robustness. For example, this happens for $k = 1$ and $s = 1$ whenever $H(\cdot)$ and $G(\cdot)$ are within 0.09 of the Gaussian nominal distribution at $x = 1$. Thus one must be cautious about designing for a certain degree of contamination if there is doubt that the contamination is really that large, particularly in the presence of a strong signal. One can compare this somewhat surprising observation with that obtained above for the parameterized case, wherein the linear detector was more robust than the censored one.

In the spirit of fairness, we should point out that conclusions drawn from the aforementioned generalized nonlocal robustness example should be regarded cautiously because the regions considered contain elements which correspond to nonmonotone functions. These "extra" elements may affect the implications derived from Theorem 2. However, our results, together with those for the parameterized case, do raise questions heretofore not sufficiently well posed to have meaning. A more subtle analysis which focusses on regions containing only elements which correspond to actual distribution functions would certainly shed more light on these questions. One question which can be directly addressed by the above work concerns the appropriateness of our local measure of robustness given by the Theorem of [2] and reproduced here as Theorem 1. We might wonder if this measure is altogether appropriate since it may reflect contributions to the slope of the surface from elements which do not correspond to distribution functions. However, the proof of Theorem 2 can be used to show that for each m there exist distribution functions $G_m(\cdot)$ and $H_m(\cdot)$ which generate a region containing the nominal distribution and **only** points which correspond to distribution functions. As the limits are taken we then obtain $\cos\gamma$ of Theorem 1, i.e. the local measure of robustness arises through our nonlocal methods applied over sets of points corresponding exclusively to distribution functions. The appropriateness of the local definition, if one interprets "local" in the sense of small regions generated by $G(\cdot)$ and $H(\cdot)$, is thus reinforced.

IV. CONCLUSION

We have applied a new approach toward robust signal detection which optimizes a cost funtion consisting of a linear combination of detector performance and robustness. This approach incorporates the geometric concept of unit normal slope for measuring the detector robustness. In addition, we have modified the local geometric methods of [2] to accomodate a greater uncertainty regarding the nominal distribution. Our results have raised some intriguing possibilities, such as situations where maximum robustness is not obtained through maximal censoring, and cases where the linear detector is more robust than the censored detector for nominally Gaussian noise.

ACKNOWLEDGEMENT

This research was supported by the Air Force Office of Scientific Research under Grant AFOSR-82-0033.

REFERENCES

1. P. J. Huber, ***Robust Statistics***, Chap. 10. New York: Wiley, 1981.
2. M. W. Thompson and D. R. Halverson, "A novel differential geometric approach toward robust signal detection",*Proceedings of the 1986 Conference on Information Sciences and Systems*, Princeton, New Jersey, March 19-21, 1986. pp. 11-16.
3. R. L. Bishop and S. I. Goldberg, *Tensor Analysis on Manifolds*, Chap. 4. New York: Macmillan, 1968.

MINIMAX ROBUST DISCRIMINATION FROM DIFFUSION PROCESSES WITH UNCERTAIN PARAMETERS

EVAGGELOS GERANIOTIS and YAWGENG A. CHAU

Department of Electrical Engineering
and Systems Research Center
University of Maryland
College Park, MD 20742

ABSTRACT

Discrimination problems characterized by observations that satisfy diffusion equations are considered. The parameters of the diffusions belong to one of the following distinct classes: (i) classes determined by 2-alternating capacities and (ii) classes with minimal or maximal elements. Both fixed observation-time-interval tests and sequential tests based on the least-favorable pairs of distributions for the aforementioned uncertainty classes are derived and shown to have an acceptable level of performance despite the uncertainty.

I. Introduction

In contrast to discrete-time robust discrimination problems, continuous-time robust discrimination problems have not received sufficient attention. In this paper we investigate both fixed observation-time-interval and sequential robust discrimination from diffusion observations whose parameters, due to modeling uncertainties or partially known noise statistics, are only known to belong to uncertainty classes.

Two distinct types of uncertainty classes are considered: (i) classes determined by two alternating capacities and (ii) classes with minimal or maximal elements. These uncertainty models have been very popular among the robust statisticians [1]-[4] and minimax robust signal processing techniques have received considerable attention in the past fifteen years (see the tutorial in [5]).

The design philosophy that we pursue is that of minimax robustness. According to it, the worst-case operational conditions are identified with regard to the error probabilities of decision designs and the optimal such design for these conditions is derived. Subsequently this decision design is employed independently of actual conditions (which are not known, except for the fact that they belong to some structured uncertainty class) and it is shown that it achieves desirable performance despite the uncertainty.

In particular, when the observation-time interval is fixed, non-homogeneous diffusions--whose time-varying parameters (means and/or variances) belong to 2-alternating capacity classes or to classes with minimal or maximal elements--are considered and the least favorable operating conditions, as well as the minimax robust likelihood ratio tests, are derived. On the other hand, when sequential tests are employed, we can only derive the least favorable operational conditions and the minimax robust sequential probability ratio test for homogeneous diffusions, whose parameters belong to classes with minimal or maximal elements. In this case, the performance criterion includes--besides the error probabilities of the test--the expected values of the stopping time under the two hypotheses.

This research was supported by the Systems Research Center at the University of Maryland, College Park, through National Science Foundation through grant CDR-85-00108.

II. Nonhomogeneous Diffusions with Fixed Observation-Time Interval

II.A Uncertainty Classes Determined by 2-Alternating Capacities

Definition: A positive set function v is called a **2-alternating capacity** if it satisfies the following conditions

(i) $v(\phi) = 0$, $v(\Omega) > 0$

(ii) $A \subset B \Rightarrow v(A) \leq v(B)$, for $A, B \in B$

(iii) $A_n \uparrow A \Rightarrow v(A_n) \uparrow v(A)$

(iv) $B_n \downarrow B \Rightarrow v(B_n) \downarrow v(B)$, on closed sets

(v) $v(A \cup B) + v(A \cap B) \leq v(A) + v(B)$

Suppose now that M is the class of measures on (Ω, B) and $m \in M$ is a measure. Consider the uncertainty class which is determined by the 2-alternating capacity v as follows

$$M_v = \left\{ m \in M \mid m(A) \leq v(A) , \; V A \in B , m(\Omega) = v(\Omega) \right\}. \tag{1}$$

When Ω is compact several popular uncertainty models like ϵ–contaminated neighborhoods [1], total variation neighborhoods [1], band classes [3] and p-point classes [4] are special cases of this model.

Example: The ϵ-contaminated model

$$M_\epsilon = \left\{ m \in M \mid m(A) = (1-\epsilon)\, m^0(A) + \epsilon \tilde{m}(A), \; V A \in B, \; m^0(\Omega) = \tilde{m}(\Omega) \right\}, \tag{2}$$

for $\epsilon \in [0, 1]$. Then $v(A) = (1-\epsilon)\, m^0(A) + \epsilon\, m^0(\Omega)$

Fundamental properties of these uncertainty models have been studied by Huber and Strassen [2]. We will state the relevant properties as a Lemma.

Lemma 1: Suppose v_0 and v_1 are 2-alternating capacities on (Ω, B) and M_0 and M_1 are the uncertainty classes determined by them as in (1). Then there exists a Lebesgue-measurable function $\pi_v : \Omega \rightarrow [0, \infty]$ such that

$$\theta\, v_0(\{\pi_v > \theta\}) + v_1(\{\pi_v > \theta\}) \leq \theta\, v_0(A) + v_1(A^c) \tag{3}$$

for all $A \in B$ and all $\theta \geq 0$. Furthermore there exist measures $(\hat{m}_0, \hat{m}_1)$ in $M_0 \times M_1$ such that

$$\hat{m}_0(\{\pi_v > \theta\}) = v_0(\{\pi_v > \theta\}) \tag{4}$$

$$\hat{m}_1(\{\pi_v \leq \theta\}) = v_1(\{\pi_v \leq \theta\}) \tag{5}$$

where π_v is a version of $d\hat{m}_1/d\hat{m}_0$ and is unique a.e. $[\hat{m}_0]$. The measures $(\hat{m}_0, \hat{m}_1)$ are termed the **least-favorable** measures over $M_0 \times M_1$.

Example: The ϵ–contaminated mixture uncertainty classes described by

$$M_j = \left\{ m_j \in M \mid m_j = (1-\epsilon_j)\, m_j^0 + \epsilon_j \tilde{m}_j , \; \tilde{m}_j(\Omega) = m_j^0(\Omega) \right\}, \; j = 0, 1 \tag{6}$$

associated with the 2-alternating capacities

$$v_j (A) = \begin{cases} (1 - \epsilon_j) m_j^0 (A) + \epsilon_j , & A \neq \phi \\ 0 & , A = \phi \end{cases} , \tag{7}$$

have the least-favorable distributions

$$d\hat{m}_0 / d\lambda = \begin{cases} (1 - \epsilon_0) \, d\hat{m}_0 / d\lambda , & dm_1^0 / dm_0^0 \leq c_2 \\ \dfrac{1 - \epsilon_0}{c_2} d\hat{m}_1 / d\lambda , & c_2 < dm_1^0 / dm_0^0 \end{cases} \tag{8}$$

$$d\hat{m}_1 / d\lambda = \begin{cases} (1 - \epsilon_1) \, d\hat{m}_0 / d\lambda , & c_1 < dm_1^0 / dm_0^0 \\ c_1 (1 - \epsilon_1) \, d\hat{m}_0 / d\lambda , & dm_1^0 / dm_0^0 \leq c_1 \end{cases} \tag{9}$$

and the Huber-Strassen derivative π_v

$$\pi_v = d\hat{m}_1 / d\hat{m}_0 = \frac{1 - \epsilon_1}{1 - \epsilon_0} \min \left\{ c_2 , \max (c_1 , dm_1^0 / dm_0^0) \right\} \tag{10}$$

where $0 \leq c_1 \leq c_2 < \infty$ are such that $\hat{m}_1 (\Omega) = \hat{m}_0 (\Omega) = 1$.

II.B Uncertainty in the Mean Functions

Consider the hypothesis testing problem

$$\begin{aligned} H_1 : \; dY_t &= M_1 (dt) + \sigma(t) \, dW_t \\ H_0 : \; dY_t &= M_0 (dt) + \sigma(t) \, dW_t \quad , 0 \leq t \leq T \end{aligned} \tag{11}$$

where W_t is a standard Wiener process and the measures M_i belong to following sets

$$\mathcal{M}_i = \left\{ M_i \in M \mid M_i (A) \leq v_i (A), \forall A \in B , M_i (\Omega) = v_i (\Omega) \right\}, \; i = 0, 1 \tag{12}$$

while the function $\sigma(t)$ is known. Here we assume that the nominal measures involved in the definition of the uncertainty classes above are absolutely continuous with respect to the Lebesgue measure λ on Ω. That is, $M_i^0 << \lambda$, $i = 0, 1$. This implies that the least-favorable measures singled out by Lemmma 1 also satisfy $\hat{M}_i << \lambda$; we define $\hat{\mu}_i (t) = (d\hat{M}_i / d\lambda)(t)$.

Proposition 1: For the hypothesis testing problem (11) and the uncertainty classes (12) the likelihood ratio test based on

$$\ln \frac{d\hat{P}_1}{d\hat{P}_0} = \int_\Omega \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\sigma(t)]^2} \, dY_t - \frac{1}{2} \int_\Omega \frac{[\hat{\mu}_1(t)]^2 - [\hat{\mu}_0(t)]^2}{[\sigma(t)]^2} \, dt \tag{13}$$

is a minimax robust test; i.e., the error probabilities for this test under the two hypotheses satisfy the inequalities

$$\alpha (\hat{\mu}_0 , \hat{\mu}_1 ; M_0) \leq \alpha (\hat{\mu}_0 , \hat{\mu}_1 ; \hat{M}_0) \tag{14}$$

$$\beta (\hat{\mu}_0 , \hat{\mu}_1 ; M_1) \leq \beta (\hat{\mu}_0 , \hat{\mu}_1 ; \hat{M}_1) \tag{15}$$

[In (14)-(15) we did not include the right-hand inequalities which state that the likelihood ratio test based on (13) is optimal for the case that $\hat{M}_i$, $i = 0, 1$ are the mean measures in (11); these missing inequalities complete the description of the saddle point].

Proof: According to Kailath' s general likelihood ratio formula [6], we have: $\ln \frac{d\hat{P}_1}{d\hat{P}_0} = \int_\Omega \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\sigma(t)]^2} \, dY_t - h$, where $h = \frac{1}{2} \int_\Omega \frac{[\hat{\mu}_1(t)]^2 - [\hat{\mu}_0(t)]^2}{[\sigma(t)]^2} \, dt$. Also define $g = \int_\Omega \left\{ \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\sigma(t)]^2} \right\}^2 [\sigma(t)]^2 \, dt > 0$, $f_i = \int_\Omega \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\sigma(t)]^2} M_i(\, dt \,)$, and $f_i = \int_\Omega \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\sigma(t)]^2} \hat{M}_i(\, dt \,)$, for $i = 0, 1$. Then

$$\alpha\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,;\,M_0\,) = P_0 \left(\ln \frac{d\hat{P}_1}{d\hat{P}_0} > \ln \eta \right) = Q \left(\frac{\ln \eta + h - f_0}{g} \right)$$

$$\leq Q \left(\frac{\ln \eta + h - \hat{f}_0}{g} \right) = \alpha\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,;\,\hat{M}_0\,) \tag{16}$$

The inequality in (16) is valid since $Q\left(-\frac{x}{c}\right)$ is for $c = g > 0$ an increasing function of x and

$$f_0 = \int_\Omega \left[\frac{\hat{\mu}_1(t)}{\hat{\mu}_0(t)} - 1 \right] \frac{\hat{\mu}_0(t)}{[\sigma(t)]^2} M_0(dt) \leq \int_\Omega \left[\frac{\hat{\mu}_1(t)}{\hat{\mu}_0(t)} - 1 \right] \frac{\hat{\mu}_0(t)}{[\sigma(t)]^2} \hat{M}_0(dt) = \hat{f}_0$$

which follows from Lemma 1 because of the stochastic dominance property (4). To prove (15) notice that

$$\beta\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,;\,M_1\,) = P_1 \left(\ln \frac{d\hat{P}_1}{d\hat{P}_0} < \ln \eta \right) = \Phi \left(\frac{\ln \eta + h - f_1}{g} \right)$$

$$\leq \Phi \left(\frac{\ln \eta + h - \hat{f}_1}{g} \right) = \beta\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,;\,\hat{M}_1\,) \tag{17}$$

and use the fact that for $c = g > 0$ $\Phi\left(-\frac{x}{c}\right)$ is a decreasing function of x and that because of Lemma 1

$$f_1 = \int_\Omega \left[\frac{\hat{\mu}_1(t)}{\hat{\mu}_0(t)} - 1 \right] \frac{\hat{\mu}_0(t)}{[\sigma(t)]^2} M_1(dt) \geq \int_\Omega \left[\frac{\hat{\mu}_1(t)}{\hat{\mu}_0(t)} - 1 \right] \frac{\hat{\mu}_0(t)}{[\sigma(t)]^2} \hat{M}_1(dt) = \hat{f}_1$$

II.C Uncertainty in the Variance

Consider the hypothesis testing problem

$$H_1 : \; dY_t = \mu_1(t)\, dt + dW_t$$

$$H_0 : \; dY_t = \mu_0(t)\, dt + dW_t \quad , 0 \leq t \leq T \tag{18}$$

where the functions $\mu_i(t)$, $i = 0, 1$ are known and the the non-standard Wiener process W_t has zero mean and variance

$$E\left\{ \left[\int_A dW_t \right]^2 \right\} = \Sigma(A), \quad \forall A \in B \,. \tag{19}$$

The measure Σ belongs to an uncertainty class of the form

$$\Sigma = \left\{ \Sigma \in M \mid \Sigma(A) \leq v(A),\ \forall A \in B,\ \Sigma(\Omega) = v(\Omega) \right\} \tag{20}$$

Here we again assume that the nominal measure(s) involved in the definition of the uncertainty class are absolutely continuous with respect to the Lebesgue measure λ on Ω. That is $\Sigma^0 \ll \lambda$ which implies that the least-favorable measure singled out by Lemma 1 (where in this case $v_1 = v$ and $v_0 = \lambda$ for the two capacities) also satisfies $\hat{\Sigma} \ll \lambda$; we define $[\hat{\sigma}(t)]^2 = (d\hat{\Sigma}/d\lambda)(t)$.

Proposition 2: For the hypothesis testing problem (18) with (19)-(20) the likelihood ratio test based on

$$\ln \frac{d\hat{P}_1}{d\hat{P}_0} = \int_\Omega \frac{\mu_1(t) - \mu_0(t)}{[\hat{\sigma}(t)]^2}\, dY_t - \frac{1}{2}\int_\Omega \frac{[\mu_1(t)]^2 - [\mu_0(t)]^2}{[\hat{\sigma}(t)]^2}\, dt\ , \tag{21}$$

is a minimax robust test; i.e., the error probabilities for this test under the two hypotheses satisfy the inequalities

$$\alpha(\hat{\sigma}; \Sigma) \leq \alpha(\hat{\sigma}; \hat{\Sigma}) \tag{22}$$

$$\beta(\hat{\sigma}; \Sigma) \leq \beta(\hat{\sigma}; \hat{\Sigma}) \tag{23}$$

[Again as for Proposition 1 we did not include the right hand-side inequalities in (22)-(23) which state that the test based on (21) is optimal for the problem (18) when W_t has variance $\hat{\Sigma}$.]

Proof: Following the same steps as in the proof for Proposition 1 we first define the quantities $h = \frac{1}{2}\int_\Omega \frac{[\mu_1(t)]^2 - [\mu_0(t)]^2}{[\hat{\sigma}(t)]^2}\, dt$, $f_i = \int_\Omega \frac{\mu_1(t) - \mu_0(t)}{[\hat{\sigma}(t)]^2}\, \mu_i(t)\, dt$, $g = \int_\Omega \left\{ \frac{\mu_1(t) - \mu_0(t)}{[\hat{\sigma}(t)]^2} \right\}^2 \Sigma(dt)$, and $\hat{g} = \int_\Omega \left\{ \frac{\mu_1(t) - \mu_0(t)}{[\hat{\sigma}(t)]^2} \right\}^2 \hat{\Sigma}(dt)$. Then we prove inequality (22) by noticing that

$$\alpha(\hat{\sigma}; \Sigma) = P_0\left(\ln \frac{d\hat{P}_1}{d\hat{P}_0} > \ln \eta \right) = Q\left(\frac{\ln \eta + h - f_0}{g} \right)$$

$$\leq Q\left(\frac{\ln \eta + h - f_0}{\hat{g}} \right) = \alpha(\hat{\sigma}; \hat{\Sigma}). \tag{24}$$

This follows from the fact that for $c = h - f_0 > 0$ $Q\left(\frac{c}{x}\right)$ is an increasing function of x and the inequality

$$g = \int_\Omega \left\{ \frac{\mu_1(t) - \mu_0(t)}{[\hat{\sigma}(t)]^2} \right\}^2 \Sigma(dt) \leq \int_\Omega \left\{ \frac{\mu_1(t) - \mu_0(t)}{[\hat{\sigma}(t)]^2} \right\}^2 \hat{\Sigma}(dt) = \hat{g}$$

which is valid because of the stochastic dominance property (5) of Lemma 1 [when applied to the case $v_0 = \lambda$ and $v_1 = v$], and the fact that the integrands inside the integrals in the inequality above are decreasing functions of $[\hat{\sigma}(t)]^2 = \frac{d\hat{\Sigma}}{d\lambda}(t)$. Finally, the inequality in (23) can be proved in a similar way.

II.D Uncertainty in both the Means and the Variance

Consider the hypothesis testing problem

$$H_1 :\ dY_t = M_1(dt) + dW_t$$

$$H_0 : \ dY_t = M_0(dt) + dW_t \ , 0 \leq t \leq T \tag{25}$$

where the measures M_i belong to classes of the form (12) and the non-standard Wiener process W_t has zero mean and variance given by (19) where the measure Si belong to uncertainty classes of the form (20). We assume that the nominal measures satisfy $M_i^0 << \lambda$ and $\Sigma^0 << \lambda$ so that the least-favorable measures $\hat{M}_i << \lambda$ and $\hat{\Sigma} << \lambda$ and we define $\hat{\mu}_i(t) = (d\hat{M}_i / d\lambda)(t)$ and $[\hat{\sigma}(t)]^2 = (d\hat{\Sigma}/d\lambda)(t)$. Then, the following result is valid.

Proposition 3: For the hypothesis testing problem (25) and the uncertainty models (12) and (20) the likelihood ratio test based on

$$\ln \frac{d\hat{P}_1}{d\hat{P}_0} = \int_\Omega \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\hat{\sigma}(t)]^2} \, dY_t - \frac{1}{2} \int_\Omega \frac{[\hat{\mu}_1(t)]^2 - [\hat{\mu}_0(t)]^2}{[\hat{\sigma}(t)]^2} \, dt , \tag{26}$$

is a minimax robust test; i.e., the error probabilities for this test under the two hypotheses satisfy the inequalities

$$\alpha (\hat{\mu}_0 , \hat{\mu}_1 , \hat{\sigma} ; M_0 , \Sigma) \leq \alpha (\hat{\mu}_0 , \hat{\mu}_1 , \hat{\sigma} ; \hat{M}_0 , \hat{\Sigma}) \tag{27}$$

$$\beta (\hat{\mu}_0 , \hat{\mu}_1 , \hat{\sigma} ; M_1 , \Sigma) \leq \beta (\hat{\mu}_0 , \hat{\mu}_1 , \hat{\sigma} ; \hat{M}_1 , \hat{\Sigma}) \tag{28}$$

[Again as for Propositions 1 and 2 we did not include the right hand-side inequalities in (27)-(28) which state that the test based on (26) is optimal for the problem (25)].

Proof: It follows from combining the proofs of Propositions 1 and 2 and is omitted.

III. Nonhomogeneous Diffusions with Parameters in Uncertainty Classes with Minimal or Maximal Elements

Consider the hypothesis testing problem

$$\begin{aligned} H_1 \ &: \quad dY_t = \mu_1(t)\, dt + \sigma(t)\, dW_t \\ H_0 \ &; \quad dY_t = \mu_0(t)\, dt + \sigma(t)\, dW_t \quad , 0 \leq t \leq T \end{aligned} \tag{29}$$

where W_t are standard Wiener processes, $\mu_i(t)$, $i = 0, 1$ belong to the classes

$$M_1 = \left\{ \mu_1(t) \ | \ \mu_1(t) \geq \hat{\mu}_1(t), 0 \leq t \leq T \right\} \tag{30}$$

$$M_0 = \left\{ \mu_0(t) \ | \ \mu_0(t) \leq \hat{\mu}_0(t), 0 \leq t \leq T \right\} , \tag{31}$$

and $\sigma(t)$ belongs to the class

$$\Sigma = \left\{ \sigma(t) \ | \ \sigma(t) \leq \hat{\sigma}(t) \ , \ 0 \leq t \leq T \right\} \tag{32}$$

For these uncertainty classes which are determined by a maximal or a minimal element the following result holds:

Proposition 4: For the hypothesis testing problem (29) and the classes (30)-(32) and under the assumption that $\hat{\mu}_1(t) \geq \hat{\mu}_0(t)$ for $0 \leq t \leq T$, the likelihood ratio test based on (26) is minimax robust, i.e., the error probabilities of the test satisfy the inequalities

$$\alpha (\hat{\mu}_0 , \hat{\mu}_1 , \hat{\sigma} ; \mu_0 , \sigma) \leq \alpha (\hat{\mu}_0 , \hat{\mu}_1 , \hat{\sigma} ; \hat{\mu}_0 , \hat{\sigma}) \tag{33}$$

$$\beta(\hat{\mu}_0, \hat{\mu}_1, \hat{\sigma}; \mu_1, \sigma) \leq \beta(\hat{\mu}_0, \hat{\mu}_1, \hat{\sigma}; \hat{\mu}_1, \hat{\sigma}) \tag{34}$$

[Again as for Propositions 1,2 and 3 we did not include the right hand-side inequalities in (33)-(34) which state that the test based on (26) is optimal for the problem (29)].

Proof: We only prove (33); (34) can be proved in a similar way. First we define $h = \frac{1}{2}\int_\Omega \frac{[\hat{\mu}_1(t)]^2 - [\hat{\mu}_0(t)]^2}{[\hat{\sigma}(t)]^2}\, dt$, $f_i = \int_\Omega \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\hat{\sigma}(t)]^2}\, \mu_i(t)\, dt$, $\hat{f}_i = \int_\Omega \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\hat{\sigma}(t)]^2}\, \hat{\mu}_i(t)\, dt$, $g = \int_\Omega \left\{ \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\hat{\sigma}(t)]^2} \right\}^2 [\sigma(t)]^2\, dt$, and $\hat{g} = \int_\Omega \left\{ \frac{\hat{\mu}_1(t) - \hat{\mu}_0(t)}{[\hat{\sigma}(t)]^2} \right\} [\hat{\sigma}(t)]^2\, dt$. Then we notice that

$$\alpha(\hat{\mu}_0, \hat{\mu}_1, \hat{\sigma}; \mu_0, \sigma) = P_0\left(\ln \frac{d\hat{P}_1}{d\hat{P}_0} > \ln \eta \right) = Q\left(\frac{\ln \eta + h - f_0}{g} \right)$$

$$\leq Q\left\{ \frac{\ln \eta + h - \hat{f}_0}{\hat{g}} \right) = \alpha(\hat{\mu}_0, \hat{\mu}_1, \hat{\sigma}; \hat{\mu}_0, \hat{\sigma}). \tag{35}$$

This follows from the fact that for $c = g > 0$ and $d = h - \hat{f}_0 > 0$ $Q\left(-\frac{x}{c}\right)$ and $Q\left(\frac{d}{y}\right)$ are increasing functions of x and y, respectively, and from the inequalities $f_0 \leq \hat{f}_0$ and $g \leq \hat{g}$ which are satisfied because of the definitions of the uncertainty classes and the assumptions stated in Proposition 4.

IV. Sequential Detection of Homogeneous Diffusions with Parameters in Uncertainty Classes with Minimal or Maximal Elements

Consider the hypothesis testing problem

$$H_1 : \quad dY_t = \mu_1\, dt + \sigma\, dW_t$$

$$H_0 : \quad dY_t = \mu_0\, dt + \sigma\, dW_t, \quad 0 \leq t \leq \tau \tag{36}$$

where τ is the stopping time and μ_1, μ_0, σ are time-invariant parameters which belong to the following uncertainty classes characterized by their minimal or maximal elements (for which $\hat{\mu}_1 \geq \hat{\mu}_0$).

$$M_1 = \left\{ \mu_1 \mid \mu_1 \geq \hat{\mu}_1 \right\}, \tag{37}$$

$$M_0 = \left\{ \mu_0 \mid \mu_0 \leq \hat{\mu}_0 \right\}, \tag{38}$$

$$\Sigma = \left\{ \sigma \mid \sigma \leq \hat{\sigma} \right\}. \tag{39}$$

In this section, the sequential probability ratio test (SPRT) [7] will be used to discriminate between H_1 and H_0. The performance criterion includes, besides the error probabilities of the test, the expected values of the stopping time under the two hypotheses. For the latter, we have the following result for the situation characterized by mismatch:

Lemma 2: Suppose τ is the stopping time of SPRT associated with (36) and the SPRT employs a likelihood ratio and thresholds which are matched to the case ($\hat{\mu}_0, \hat{\mu}_1, \hat{\sigma}$) while the actual operating conditions are determined [through (36)] by (μ_0, μ_1, σ). Then the following identities are true

$$E_0[\tau \mid \hat{L}] = \frac{-2\,\hat{\sigma}^2\,\omega(\hat{\alpha}, \hat{\beta}, \alpha)}{(\hat{\mu}_1-\hat{\mu}_0)(2\mu_0 - \hat{\mu}_1 - \hat{\mu}_0)} \tag{40}$$

$$E_1[\tau \mid \hat{L}] = \frac{2\,\hat{\sigma}^2\,\omega(\hat{\beta}, \hat{\alpha}, \beta)}{(\hat{\mu}_1-\hat{\mu}_0)(2\mu_1 - \hat{\mu}_1 - \hat{\mu}_0)} \tag{41}$$

where

$$\omega(\hat{x}, \hat{y}, x) = (1-x)\ln\frac{1-\hat{x}}{\hat{y}} + x\ln\frac{\hat{x}}{1-\hat{y}} \tag{42}$$

Proof: Under the actual operation conditions, we have

$$E_0\left[\ln\frac{d\hat{P}_1}{d\hat{P}_0}\right] = E_0\left[\frac{\hat{\mu}_1-\hat{\mu}_0}{\hat{\sigma}^2}y_\tau - \frac{\hat{\mu}_1^2-\hat{\mu}_0^2}{2\hat{\sigma}^2}\tau\right] = \frac{(\hat{\mu}_1-\hat{\mu}_0)(2\mu_0-\hat{\mu}_1-\hat{\mu}_0)}{2\hat{\sigma}^2}E_0[\tau \mid \hat{L}] \tag{43}$$

Furthermore

$$E_0\left[\ln\frac{d\hat{P}_1}{d\hat{P}_0}\right] = \int_\Omega \ln\frac{d\hat{P}_1}{d\hat{P}_0}\,dP_0 = \int_{H_1}\ln\frac{d\hat{P}_1}{d\hat{P}_0}\,dP_0 + \int_{H_0}\ln\frac{d\hat{P}_1}{d\hat{P}_0}\,dP_0$$

$$= \alpha\ln\frac{1-\hat{\beta}}{\hat{\alpha}} + (1-\alpha)\ln\frac{\hat{\beta}}{1-\hat{\alpha}} = -\,\omega(\hat{\alpha},\hat{\beta},\alpha) \tag{44}$$

where ω was defined by (42). Then (40) follows from (43) and (44). A similar proof holds for (41). For the probabilities of error α and β, the following results can be obtained under mismatch:

Lemma 3: Suppose $\hat{A}$ and $\hat{B}$ ($0 < \hat{A} < \hat{B}$) are the thresholds of the SPRT with respect to hypotheses (36) which are matched to the case ($\hat{\mu}_0$, $\hat{\mu}_1$, $\hat{\sigma}$). If the actual operating conditions are determined by (μ_0 , μ_1 , σ), then the following identities hold

$$\alpha(\hat{\mu}_0, \hat{\mu}_1, \hat{\sigma}; \mu_0, \sigma) = \frac{\hat{A}^{r_0} - 1}{(\hat{B}^{-1}\hat{A})^{r_0} - 1} \tag{45}$$

$$\beta(\hat{\mu}_0, \hat{\mu}_1, \hat{\sigma}; \mu_1, \sigma) = \frac{\hat{B}^{r_1} - 1}{(\hat{A}^{-1}\hat{B})^{r_1} - 1} \tag{46}$$

where

$$r_0 = \frac{\hat{\sigma}^2(2\mu_0 - \hat{\mu}_1 - \hat{\mu}_0)}{\sigma^2(\hat{\mu}_1 - \hat{\mu}_0)} \tag{47}$$

and

$$r_1 = \frac{\hat{\sigma}^2(2\mu_1 - \hat{\mu}_1 - \hat{\mu}_0)}{\sigma^2(\hat{\mu}_1 - \hat{\mu}_0)}\,. \tag{48}$$

Proof: Let $\rho = (\hat{\mu}_1 - \hat{\mu}_0)/\hat{\sigma}^2$, then under the assumption $Y_0 = 0$, we have

$$\alpha = P_0\left(\ln\frac{d\hat{P}_1}{d\hat{P}_0}\text{ hits }\ln\hat{B}\text{ before }\ln\hat{A}\right) = P_0\left(Y_t - \frac{\hat{\mu}_1+\hat{\mu}_0}{2}t\text{ hits }\frac{\ln\hat{B}}{\rho}\text{ before }\frac{\ln\hat{A}}{\rho}\right) \tag{49}$$

Let $Z_t = Y_t - \dfrac{\hat{\mu}_1+\hat{\mu}_0}{2}t$, then Z_t has under H_0 mean $\mu_0 - \dfrac{\hat{\mu}_1+\hat{\mu}_0}{2}$ and variance $\sigma^2 t$. Next, we write the false alarm probability α as [8]

$$\alpha = E\ [\, P_0(\, Z_h\,)\,] + o\,(\, h\,) \tag{50}$$

where $o\ (\ h\ )$ denotes the probability that the process would have already hit one of the barriers by the time h . Expanding $P_0\ (\ Z_h\ )$ around $Z_0 = 0$, we obtain

$$\alpha = E\ [\, \alpha + \dot{\alpha}\, Z_h + \ddot{\alpha}\, Z_h^2/2 + \ \cdots\ ] + o\,(\, h\,) \tag{51}$$

where $\dot{\alpha}$, $\ddot{\alpha}$, $\cdots$ denote derivatives with respect to Z_h evaluated at $Z_h = Z_0 = 0$. Let $h \rightarrow 0$, then the following differential equation will result

$$\dot{\alpha}\ \frac{\hat{\sigma}\,(\, 2\,\mu_0 - \hat{\mu}_0 - \hat{\mu}_1\,)}{2\,\sigma^2\,(\,\hat{\mu}_1 - \hat{\mu}_0\,)} + \frac{\ddot{\alpha}}{2} = 0 \tag{52}$$

By solving this differential equation and using the appropriate boundary values for the two barriers, we obtain

$$\alpha = \frac{\hat{A}^{\,r_0} - 1}{(\hat{B}^{-1}\hat{A}\,)^{r_0} - 1} \tag{53}$$

A similar proof can be developed for $\beta = \beta\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,,\,\hat{\sigma}\,;\,\mu_1\,,\,\sigma\,)$.
Equipped with the above two lemmas, we now prove the main result of this section.

Proposition 5: Consider the hypothesis testing problem of (36) with parameters in the sets (37)-(39). Suppose that the following assumptions are satisfied:

$$(i)\quad \hat{\mu}_1 \geq \hat{\mu}_0 \tag{54}$$

$$(ii)\quad \ln \frac{1-\hat{\beta}}{\hat{\alpha}} >> -\,\hat{\beta}\ln \frac{\hat{\alpha}\,\hat{\beta}}{(\,1-\hat{\alpha}\,)\,(\,1-\hat{\beta}\,)}\ . \tag{55}$$

Then the following inequalities are true

$$\alpha\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,,\,\hat{\sigma}\,;\,\mu_0\,,\,\sigma\,) \leq \alpha\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,,\,\hat{\sigma}\,;\,\hat{\mu}_0\,,\,\hat{\sigma}\,) \tag{56}$$

$$\beta\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,,\,\hat{\sigma}\,;\,\mu_1\,,\,\sigma\,) \leq \hat{\beta}\,(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,,\,\hat{\sigma}\,;\,\hat{\mu}_1\,,\,\hat{\sigma}\,) \tag{57}$$

$$E_i\ [\,\tau \mid \hat{L}\,] \leq \hat{E}_i\ [\,\tau \mid \hat{L}\,]\ ,\quad i = 0,\,1\ . \tag{58}$$

Proof: First we show (58) for $i = 1$. From Lemma 2 and because of assumptions (ii) and (i) we have

$$\begin{aligned} E_1\,[\,\tau \mid \hat{L}\,] &= \frac{2\,\hat{\sigma}^2}{(\,\hat{\mu}_1 - \hat{\mu}_0\,)\,(\,2\,\mu_1 - \hat{\mu}_1 - \hat{\mu}_0\,)} \left[\,(\,1-\beta\,)\ln \frac{1-\hat{\beta}}{\hat{\alpha}} + \beta \ln \frac{\hat{\beta}}{1-\hat{\alpha}}\,\right] \\ &\approx \frac{2\,\hat{\sigma}^2}{(\,\hat{\mu}_1 - \hat{\mu}_0\,)\,(\,2\,\mu_1 - \hat{\mu}_1 - \hat{\mu}_0\,)} \ln \frac{1-\hat{\beta}}{\hat{\alpha}} \leq \frac{2\,\hat{\sigma}^2}{(\,\hat{\mu}_1 - \hat{\mu}_0\,)^2} \ln \frac{1-\hat{\beta}}{\hat{\alpha}} \\ &\approx \frac{2\,\hat{\sigma}^2}{(\,\hat{\mu}_1 - \hat{\mu}_0\,)^2} \left[\,(\,1-\hat{\beta}\,)\ln \frac{1-\hat{\beta}}{\hat{\alpha}} + \hat{\beta} \ln \frac{\hat{\beta}}{1-\hat{\alpha}}\,\right] = \hat{E}_1\,[\,\tau \mid \hat{L}\,] \end{aligned} \tag{59}$$

where the first equality follows from (43), the inequality follows from the fact that $\mu_1 \leq \hat{\mu}_1$, the two approximations are valid because of assumption (ii), and the second equality follows from (43) for $\mu_1 = \hat{\mu}_1$. A similar proof can be given for $E_0\ [\ \tau \mid \hat{L}\ ]$ [(58) for $i = 0$].
Next we prove (56). From Lemma 3 we notice that α depends on $(\,\hat{\mu}_0\,,\,\hat{\mu}_1\,,\,\hat{\sigma}\,;\,\mu_0\,,\,\sigma\,)$ only through r_0. Thus we have

$$\frac{d\alpha}{dr_0} = \frac{(\hat{B}^{-1}\hat{A})^{r_0}}{\left((\hat{B}^{-1}\hat{A})^{r_0} - 1\right)^2} \left[(1 - \hat{B}^{r_0}) \ln\hat{A} + (\hat{A}^{r_0} - 1) \ln\hat{B} \right]. \tag{60}$$

The sign of this derivative is determined by $(1 - \hat{B}^{r_0}) \ln\hat{A} + (\hat{A}^{r_0} - 1) \ln\hat{B}$. At $r_0 = -1$ this becomes $\left(1 - \frac{\hat{\alpha}}{1 - \hat{\beta}}\right) \ln\frac{\hat{\beta}}{1 - \hat{\alpha}} + \left(\frac{1 - \hat{\alpha}}{\hat{\beta}} - 1\right) \ln\frac{1 - \hat{\beta}}{\hat{\alpha}}$ which is positive. From (47) for $\mu_0 < \hat{\mu}_0$ and $\sigma < \hat{\sigma}$ we can show that

$$r_0 \leq -1 \tag{61}$$

where equality is achieved for $(\mu_0, \sigma) = (\hat{\mu}_0, \hat{\sigma})$. Since α is an increasing function of r_0 at -1, its value at any r_0 smaller than -1 (i.e., for any μ_0 and σ) will be smaller than its value at $r_0 = -1$; which proves (56). A similar proof can be developed for (57); for that we need the fact that $r_1 \geq 1$.

REFERENCES

[1] P. J. Huber, "A robust version of the probability ratio test," *Ann. Math. Stat.*, vol. 36, pp. 1753-1758, 1965.

[2] P. J. Huber and V. Strassen, "Minimax tests and the Neyman-Pearson lemma for capacities," *Ann. Statist.*, vol. 1, pp. 251-265, 1973.

[3] S. A. Kassam, "Robust hypothesis testing for bounded classes of probability densities," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 242-247, 1981.

[4] K. S. Vastola and H. V. Poor, "On the p-point uncertainty class," *IEEE Trans. Information Theory*, vol. IT-30, pp.374-376, 1984.

[5] S. A. Kassam and H. V. Poor, "Robust techniques for signal processing," *Proc. IEEE*, vol. 73, pp. 433-481, March 1985.

[6] T. Kailath, "The innovations approach to detection and estimation theory," *Proc. of IEEE*, vol. 58, pp. 680-695, May 1970.

[7] A. Wald, *Sequential Analysis*, Wiley, New York, 1947.

[8] S. M. Ross, *Stochastic Processes*.

DETECTION OF WEAK SIGNALS IN A GENERALIZED OBSERVATION MODEL

IICKHO SONG and **SALEEM A. KASSAM**
Department of Electrical Engineering
University of Pennsylvania
Philadelphia, PA 19104

ABSTRACT: A generalized observation model for signal detection problems is proposed. The model includes several interesting and common special cases such as those describing additive noise, multiplicative noise, and signal-dependent noise. Locally optimum detectors for the proposed model are derived and analyzed for performance. It is shown that the locally optimum detectors are interesting generalizations of those for the additive noise model. Under the proposed observation model, performance of the locally optimum detectors is compared to that of other common detectors.

1. INTRODUCTION

The additive noise model, in which noise is added to a signal (deterministic or random) to generate observations, has been one of the most common models in many areas of signal processing such as signal detection, estimation and filtering. One reason for the popularity of the additive noise model is that it is relatively easy to treat mathematically and to obtain for it explicit and appealing structures for signal processors in various applications. Another reason for the popularity of the additive noise model is that the contributions of other processes such as multiplicative noise and signal-dependent noise may be assumed to be negligible in many situations so that the additive noise model, which is an approximation of actual circumstances, usually produces quite acceptable results. However, if it is unreasonable to assume that the additive noise model yields a good approximation for the mechanism generating the observations in a problem under consideration, then the penalty which results from using the incorrect (additive noise) model may be significant. Non-additive noise models are necessary in several different types of situations; for example, multiplicative and signal-dependent noise terms cannot be ignored in certain communications, acoustics, and image processing applications [1-4]. The effects of delayed signals from multipath or reverberation phenomena, and the actions of automatic gain control (AGC) circuits or of nonlinearities acting on additive signal and noise components may all be modeled using multiplicative and signal-dependent as well as purely additive noise terms. For this reason and more basically as a natural extension of the additive noise models in signal processing theory, it is of interest to study more general noisy signal models which include the additive noise models as special cases.

In this paper we will be concerned with detection of known signals in observations modeled by a generalized noisy signal model which accommodates multiplicative and signal-dependent noise terms in addition to purely additive noise. In particular, we will consider the locally optimum detection of known signals, based on the generalized version of the Neyman-Pearson fundamental lemma [5] of statistical hypothesis testing. Locally optimum (LO) detectors, which are optimum in detecting signals in the local case of weak signals (SNR→0), have been of considerable interest because detection of weak signals is more difficult than detection of

---

This research is supported by the Air Force Office of Scientific Research under Grant 82-0022.

signals which are comparable in strength to the noise process. Even though much effort has been devoted to the problem of LO detection of signals, most of the previously reported studies of locally optimum detection [6-10] in non-Gaussian or Gaussian noise environments have dealt only with the case of purely additive noise.

2. GENERALIZED OBSERVATION MODEL FOR NOISY SIGNAL

The multiple-input additive noise model for observations X_{ji} in the signal detection problem can be described by

$$X_{ji} = \theta Q_i + W_{ji}, \quad j=1,2,\ldots,L, \; i=1,2,\ldots,n. \tag{1}$$

Here L is the number of input channels from which observations are taken; n is the size of the sample collected at each input channel; Q_i is the common signal component at each channel at the i-th sampling instant; θ is the signal amplitude, which may be zero (null-hypothesis) or non-zero (alternative hypothesis); and W_{ji} is the noise in the j-th channel at the i-th sampling instant. In general we may have $Q_i=e_i+S_i$ where $E\{Q_i\}=e_i$ and the S_i are zero-mean random variables. The noise components W_{ji} will be taken to be independent and identically distributed (i.i.d.) random variables with common probability density function (pdf) f_W and means zero and variances V_W, with the signal and noise being statistically independent.

We will now introduce a more flexible model capable of yielding more realistic descriptions in a broader range of situations. Instead of the purely additive signal and noise model of (1), consider the model describing the observation X_{ji}, for i=1,2,...,n and j=1,2,...,L, by

$$X_{ji} = \alpha(\tau)e_i+\beta(\tau)S_i+\gamma(\tau)^{1-d}[\alpha(\tau)e_i+\beta(\tau)S_i]^d \, N_{ji}+W_{ji}. \tag{2}$$

We retain the assumptions of our additive observation model (1), but have now introduced the additional noise components N_{ji}. These are i.i.d. random variables independent of the S_i but with N_{ji} and W_{ji} generally correlated. Notice that the signal term θQ_i in (1) has been more generally broken up into deterministic and zero-mean random components with respective amplitudes $\alpha(\tau)$ and $\beta(\tau)$. This signal term multiplies N_{ji} when d=1 to produce an additional multiplicative noise term generally correlated with the purely additive component W_{ji}. On the other hand, with d=0 we get the additional term $\gamma(\tau)N_{ji}$ which is a signal-dependent noise term, since τ controls the signal strength through $\alpha(\tau)$ and $\beta(\tau)$. We will assume that $\alpha(\tau)=\beta(\tau)=\gamma(\tau)=0$ at $\tau=0$. Let the variance of the N_{ji} be V_N and let f_N be the common pdf of the N_{ji}. Finally we will denote by f_{NW} the common joint pdf of the (N_{ji}, W_{ji}).

In this paper, we will focus on the case of i.i.d. noise terms in both time and across the L inputs, and will restrict d to be either 0 or 1. We will also consider only the known signal detection problem in the above generalized observation model for noisy signal; that is, $\beta(\tau)$ will be assumed to be 0 throughout this paper.

Our detection problem can now be formulated as a statistical hypothesis testing problem of choosing between a null-hypothesis H and an alternative hypothesis K describing the joint pdf $f(\mathbf{x}|\tau)$ of the observation matrix $X=\{X_{ji}\}$, where $\mathbf{x}$ is a realization of X. Under H we have $\tau=0$ and under K we have $\tau>0$. Only $\tau>0$ is assumed to be unknown under the alternative hypothesis; $\tau=0$ in $f(\mathbf{x}|\tau)$ yields the null hypothesis pdf.

Assumptions

In computing the test statistics of LO detectors in Section 3, the following assumptions will be made:

(A) The function $\alpha(\tau)$ is strictly increasing at the origin and if d=0 the function $\gamma(\tau)$ is also strictly increasing at the origin.

(B) When d=0, let

$$\lim_{\substack{\tau\to0\\ \tau>0}} \frac{\gamma(\tau)}{\kappa\tau^p\,\alpha(\tau)} = 1, \tag{3}$$

for some $\kappa>0$ and finite p.

(C) When d=0, if $p\geq0$, let $\theta=\alpha(\tau)$ and define $a(\theta)=\theta$, $c(\theta)=\gamma(\tau)$, and if $p<0$, let $\theta=\gamma(\tau)$ and define $a(\theta)=\alpha(\tau)$, $c(\theta)=\theta$. For $p\geq0$, $c'(\theta)$ exists at the origin; for $p<0$, $a'(\theta)$ exists at the origin. When d=1 let $a(\theta)=\theta=\alpha(\tau)$.

(D) The following quantities all exist and are finite:

$$I_1(f_W) = \int g_1(x)^2 f_W(x)\,dx, \tag{4}$$

$$I_2(f_W) = \int g_1(x)g_2(x)\,f_W(x)\,dx, \tag{5}$$

$$I_3(f_W) = \int g_2(x)^2 f_W(x)\,dx, \tag{6}$$

where

$$g_1(x) = -\,f_W'(x)/f_W(x), \tag{7}$$

$$g_2(x) = -\,u'(x)/f_W(x), \tag{8}$$

and

$$\begin{aligned} u(x) &= \int n f_{NW}(n,x)\,dn \\ &= f_W(x)E\{N|W=x\}. \end{aligned} \tag{9}$$

Note that $I_1(f_W)$ is the Fisher's information function [11]. We will be assuming in the following development that the pdfs f_N, f_W and f_{NW} are smooth enough and satisfy regularity conditions justifying mathematical operations such as interchanges in the order of differentiations with respect to a parameter and integral of a function.

3. LOCALLY OPTIMUM DETECTORS IN THE GENERALIZED OBSERVATION MODEL

We will concentrate on the problem of designing detectors for weak signals, which results in locally optimum (LO) detectors. LO detectors generally offer the advantages of simple structures, making them useful in practice. Having a basis in the generalized Neyman-Pearson lemma [5], an LO detector has a maximum slope of the detector power function at the origin ($\theta=0$) in the class of all detectors which have its false-alarm probability. The power of the LO detector is guaranteed to be larger than that of other detectors at least for θ in some non-null interval $(0,\theta_M)$.

For known-signal detection in the additive noise model, which corresponds to the case of $V_N=0$ in the generalized observation model, it can easily be shown that the LO detector test statistic [7] is a special case of the generalized correlator array (GCA) detector test statistic which can be expressed as

$$T_{GCA}(X) = \sum_{i=1}^{n} e_i \sum_{j=1}^{L} g(X_{ji}). \tag{10}$$

More specifically, if $g(x)=g_1(x)=-f_W'(x)/f_W(x)$, (10) represents the LO detector test statistic in the additive noise model [10].

Locally Optimum Detector Test Statistic

Applying the generalized Neyman-Pearson lemma [5], we find that for most cases of interest the LO detector test statistic is

$$T_{LO}(X) = \sum_{i=1}^{n} \sum_{j=1}^{L} \{a'(0)e_i g_1(X_{ji}) + A_i g_2(X_{ji})\}, \tag{11}$$

where

$$A_i = \begin{cases} c'(0) & \text{if } d=0, \\ a'(0)e_i & \text{if } d=1. \end{cases} \tag{12}$$

The test statistic of (11) is correct for LO detection in the generalized observation model when $d=1$ and also when $d=0$ and N and W are correlated. When $d=0$ and N, W are uncorrelated the result is still valid for $p\geq 0$. If, however, $E\{N|W=x\}=0$ when $d=0$ and $p<0$, (11) has to be modified. Since the case of uncorrelated N and W is less interesting in our context than that of correlated N and W, we do not give the result for uncorrelated N and W here.

In (11) the first term containing g_1 is the same as the term in (10); the second term represents the effect of signal-dependent or multiplicative noise on the test statistic. It should be noticed that even when $E\{N\}=0$, $g_2(x)$ does not in general vanish if N and W are correlated.

Observations and Comments

(a) If $p>0$ (for example, $\alpha(\tau)=\tau$ and $\gamma(\tau)=\tau^2$) and $d=0$, only the first term that contains g_1 under the summations remains in the test statistic, because $a'(0)=1$ and $c'(0)=0$. This implies that in such circumstances the test statistic depends only on the known signal term $\alpha(\tau)e_i$ and that the signal-dependent noise term $\gamma(\tau)N_{ji}$ has no effect on the LO detector test statistic. This implies in turn that the LO detector test statistic is exactly the same as that which would be obtained for $V_N=0$ or the additive noise model. The case $p>0$ in (3) describes the situation in which the signal-dependent noise term is relatively unimportant in the generalized observation model.

(b) On the contrary, if $p<0$ and $d=0$, only the second term, containing $g_2(x_{ji})$, remains under the summations in (11). This is because $a'(0)=0$ and $c'(0)=1$ in this case. The situation is very similar to the problem of detecting a random signal in additive noise with the signal being correlated with the additive noise and the random signal strength being determined by $\gamma(\tau)$. Note that g_2 is not identically zero as long as the $E\{N|W\}$ is not zero; this condition holds for correlated N and W.

(c) When $p=0$ and $d=0$, the terms containing $g_1(x_{ji})$ and $g_2(x_{ji})$ both exist in (11) and the relative weights of the effect of the known-signal term and the signal-dependent noise term on the test statistic are determined by $a'(0)e_i$ and $c'(0)$, in which at least one of $a'(0)$ and $b'(0)$ is equal to 1 with the other being a finite number.

(d) If $e_i=0$, $i=1,2,\dots,n$ for $d=0$, which is not in a strict sense a problem of known-signal detection we can still use the statistic (11) as the basis of a detector if the two noise processes N and W are correlated so that $g_2(x)$ is not identically equal to zero.

(e) If $d=1$, the test statistic always depends on the known signal and the multiplicative noise term also has an effect on the LO detector test statistic through $e_i g_2(x)$, unless $g_2(x)$ is identically zero, which occurs if $E\{N|W=x\}=0$ for all x.

(f) It is clear that the LO detector test statistic depends on the strengths, $a(\theta)$ and $c(\theta)$, through their derivatives evaluated at $\theta=0$, and not on the exact form of the strength functions. For example, the LO detector with $a(\theta)=\theta$ and $c(\theta)=\theta$ uses exactly the same test statistic as that with $a(\theta)=\theta$ and $c(\theta)=\exp(\theta)$ or with $a(\theta)=\theta$ and $c(\theta)=\theta+\theta^3$. This is because θ, $a(\theta)$ and $c(\theta)$ tend to zero under the alternative hypothesis in the same way for these functions.

Examples

If we take a correlated bivariate Gaussian pdf $N(0,0,1,s,r)$ for f_{NW}, for which $E\{N\}=E\{W\}=0$, $V_W=1$ and $V_N=s^2$, we find that the LO detector test statistic is, assuming $a'(0)=1$, $e_i=1$, $i=1,2,\dots,n$, and $c'(0)=1$ when $d=0$,

$$T_{LO}(X) = \sum_{i=1}^{n} \sum_{j=1}^{L} \{X_{ji} + rs(X_{ji}^2-1)\}, \qquad (13)$$

where r is the correlation coefficient between the two noise processes N and W. The second term, $rs(X_{ji}^2-1)$, is quite interesting, since the square-law characteristic arises generally in zero-mean random-signal detection in purely additive noise. This term is a consequence of the randomness of the noise N, which can also be regarded as a random signal component with strength $\alpha(\tau)$ if d=1 and $\gamma(\tau)$ if d=0 in the generalized observation model. The fundamental difference, however, is that while the square-law characteristic in the LO detector for random-signal detection is a result of the second derivative operation on the noise pdf, the square-law characteristic in (13) is a consequence of the correlation between the two noise processes N and W; that is, $f_W(x)$ is multiplied by $E\{N|W=x\}=rsx$, differentiated once and then divided by $f_W(x)$ to give $g_2(x)$, during which operation the square term is produced. Figure 1 shows plots of the LO nonlinearity $g_{LO}(x)=x+rs(x^2-1)$ for various values of the product rs.

As another example, let us consider a bivariate t-distribution $t(m_1,m_2,s_1,s_2,r,k)$ whose joint pdf is given by [12]

$$f(x,y) = \frac{1}{2\pi s_1 s_2 \sqrt{1-r^2}} \left[1+ \frac{1}{k(1-r^2)} \left\{\frac{(x-m_1)^2}{s_1^2} - \frac{2rxy}{s_1 s_2} + \frac{(y-m_2)^2}{s_2^2} \right\}\right]^{-(k+2)/2}. \quad (14)$$

Here $E\{X\}=m_1$, $E\{Y\}=m_2$, $V\{X\}=s_1^2 k/(k-2)$, $V\{Y\}=s_2^2 k/(k-2)$, r is the correlation coefficient and k is a parameter that determines the rate of decay of the pdf; if k=2 (14) represents a bivariate Cauchy pdf and for $k\to\infty$ (14) tends to a bivariate Gaussian pdf.

If we take $t(0,0,1,s,r,k)$ for f_{NW}, the LO detector test statistic is now, again assuming $a'(0)=c'(0)=1$ and $e_i=1$, $i=1,2,\ldots,n$,

$$T_{LO}(X) = \sum_{i=1}^{n} \sum_{j=1}^{L} \frac{(k+1)X_{ji} + rsk(X_{ji}^2-1)}{k + X_{ji}^2} . \quad (15)$$

For the bivariate t-distribution function also the LO detector test statistic has an additional term with a square-law characteristic in the numerator, which represents the effect of the signal-dependent (if d=0) or multiplicative (if d=1) noise term. Plots of the LO nonlinearity g_{LO} for several combinations of pdf parameters are given in Figure 2. From Figures 1 and 2, the following observations can generally be made.

(1) For r>0, observations with large absolute values ($|X|>1$) are taken to be more indicative of signal presence in the generalized observation model than they would have been in the additive noise model. This effect is more pronounced as the product rs becomes larger. This is because of the third term $c(\theta)^{1-d}[a(\theta)e_i]^d N_{ji}$ in (2); under the alternative hypothesis, this term makes the absolute value of an observation larger in the generalized observation model than in the additive noise model, because N and W are correlated and r>0. The reverse effect is observed for r<0.

(2) On the other hand, observations with small absolute values ($|X|<1$) are weighted less in favor of the alternative hypothesis in the generalized observation model than they would have been weighted in the additive noise model, for r>0. This effect is again more pronounced as the

product rs becomes larger. Again, the reason for this is that under the alternative hypothesis the third term in (2) tends to make the absolute value of an observation in the generalized observation model larger if $r>0$ and smaller if $r<0$ than it would be in the additive noise model.

(3) The signal-dependent and multiplicative noise term has an influence on the test statistic through the product rs, not through individual values of r and s; thus $r=1$ and $s=0.5$ has the same effect on the test statistic as does $r=0.5$ and $s=1$. Of course, if $r=0$ (N and W are uncorrelated) or $s=0$ (purely additive noise) then the test statistic becomes that which would have been obtained in the purely additive noise model.

4. ASYMPTOTIC PERFORMANCE OF THE LOCALLY OPTIMUM DETECTORS

In comparing asymptotic performance of two detectors, the asymptotic relative efficiency (ARE) is generally employed. The ARE of a detector D_1 with respect to another detector D_2 may be defined by [13]

$$ARE_{1,2} = \lim_{\substack{n_1 \to \infty \\ n_2 \to \infty}} \frac{n_2}{n_1} , \tag{16}$$

where n_1 and n_2 are the sample sizes of the detectors D_1 and D_2, respectively, to attain a certain power for a specified false-alarm probability. Under some regularity conditions [13] $ARE_{1,2}$ can explicitly be expressed as

$$ARE_{1,2} = \xi_1/\xi_2 , \tag{17}$$

where ξ_1 and ξ_2 are called the efficacies of the detectors D_1 and D_2, respectively. The efficacies of the LO detector, the linear correlator array (LCA) detector, the sign correlator array (SCA) detector and the locally optimum detector in the purely additive noise model (LOA detector) can be calculated to be [14], assuming $\overline{a'(0)\neq 0}$,

$$\xi_{LO} = L \{a'(0)^2 \overline{e^2} I_1 + \overline{A^2} I_3 + 2a'(0)\overline{eA} I_2\}, \tag{18}$$

$$\xi_{LCA} = La'(0)^2 \overline{e^2}/V_W , \tag{19}$$

$$\xi_{SCA} = 4Lf_W(0)^2\{a'(0)\overline{e^2} + \overline{eA}\, E_0\}^2/ \overline{e^2}, \tag{20}$$

and

$$\xi_{LOA} = L \{a'(0)\, \overline{e^2} I_1 + \overline{eA}\, I_2\}^2/ \overline{e^2}\, I_1, \tag{21}$$

where

$$\overline{e^k A^m} = \lim_{n\to\infty} \sum_{i=1}^{n} e_i{}^k A_i{}^m /n, \tag{22}$$

and

$$E_0 = E\{N|W=0\}. \tag{23}$$

These efficacies are valid for both d=0 and d=1. Note that these efficacies increase linearly with L and the terms containing I_2, I_3 or E_0 represent the effect of the correlation between the two noise processes N and W and the effect of signal-dependent or multiplicative noise term; if $V_N=0$ or $E\{N|W\}=0$, these terms will become zero and the resulting expression of the efficacies will be exactly the same as that which would be obtained in the additive noise model. ARE's of the LO detector with respect to other detectors can now be found from the efficacies. For example, if we take a bivariate Gaussian pdf N(0,0,1,s,r) for f_{NW} we have

$$ARE_{LO,LCA} = 1+2r^2s^2, \tag{24}$$

$$ARE_{LO,SCA} = \pi(1+2r^2s^2)/2, \tag{25}$$

and

$$ARE_{LO,LOA} = 1+2r^2s^2, \tag{26}$$

where it is assumed that $a'(0)=c'(0)=1$ and $e_i=1$, $i=1,2,\ldots,n$. Figure 3 shows these ARE's as functions of rs. As another example, if we take t(0,0,1,s,r,k) for f_{NW} we have

$$ARE_{LO,LCA} = k(k+1+2kr^2s^2)/(k-2)(k+3), \tag{27}$$

$$ARE_{LO,SCA} = k(k+1+2kr^2s^2)\ B^2(1/2,k/2)/4(k+3), \tag{28}$$

and

$$ARE_{LO,LOA} = 1 + 2kr^2s^2/(k+1), \tag{29}$$

where B(x,y) is the beta function and the same assumptions as before are made. These ARE's are shown in Figure 4. It is observed that as the product rs becomes larger, the effect of the signal-dependent or multiplicative noise on the asymptotic performance of a detector also becomes significant.

5. SUMMARY

In this paper, a generalized observation model for problems of detecting signals from observations corrupted by various types of noise has been introduced as an interesting generalization of the purely additive noise model.

The generalized observation model introduced in this paper is a natural generalization of the purely additive noise model to accommodate the effect of multiplicative or signal-dependent noise in addition to the purely additive noise. The generalized observation model, therefore, includes simpler models such as the purely additive noise model as special cases.

Locally optimum detectors which were derived for this observation model were shown to be generalized versions of those which would for the additive noise model. It was observed that they reduced to the locally optimum detectors in the additive noise model when parameters are appropriately chosen.

Comparisons of performance of the locally optimum detectors and other detectors were also made. It has been shown that the locally optimum detectors generally significantly outperform other simpler detectors such as the linear correlator array detector and the sign correlator array detector in signal-dependent or multiplicative noise.

REFERENCES

[1] V.M. Albers, Underwater Acoustics Handbook-II, The Penn. State Univ. Press, 1965.
[2] A.V. Oppenheim, R.W. Schafer and T.G. Stockham Jr., "Nonlinear Filtering of Multiplied and Convolved Signals", IEEE Proc., vol. 56, pp. 1264-1291, Aug. 1968.
[3] R. Pettai, Noise in Receiving Systems, John Wiley & Sons, New York, 1984.
[4] D.T. Kuan, et al, "Adaptive Noise Smoothing Filter for Images with Signal Dependent Noise", IEEE Tr. Pattern Anal. Machine Intelli., vol PAMI-7, pp. 165-177, March 1985.
[5] E.L. Lehmann, Testing Statistical Hypotheses, John Wiley & Sons, New York, 1959.
[6] D. Middleton, "Canonically Optimum Threshlod Detection", IEEE Tr. Inform. Theory, vol. IT-12, pp. 230-243, March 1966.
[7] J.H. Miller and J.B. Thomas, "Detectors for Discrete-Time Signals in Non-Gaussian Noise", IEEE Tr. Inform. Theory, vol. IT-18, pp. 241-250, March 1972.
[8] L.M. Nirenberg, "Low SNR Digital Communication over Certain Additive Non-Gaussian Channels", IEEE Tr. Comm., vol. COM-23, pp. 332-341, 1975.
[9] J.W. Modestino and A.Y. Ningo, "Detection of Weak Signals in Narrowband Non-Gaussian Noise", IEEE Tr. Inform. Theory, vol. IT-25, pp. 592-600, Sept. 1979.
[10] S.A. Kassam, Signal Detection in Non-Gaussian Noise, Springer Verlag, 1987.
[11] J. Hajek and Z. Sidak, Theory of Rank Tests, Academic, 1967.
[12] N.L. Johnson and S. Kotz, Distributions in Statistics: Continuous Multivariate Distributions, John Wiley & Sons, New York, 1972.
[13] M.G. Kendall and A. Stuart, The Advanced Theory of Statistics, vol. 2, Hafner Publishing, New York, 1927.
[14] I. Song, Nonlinear Techniques for Detection and Filtering of Discrete-time Signals, Ph.D. Dissertation, Dept. of Elect. Engr., Univ. of Pennsylvania, Dec. 1986.

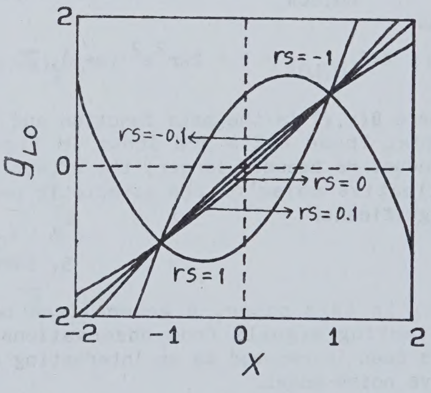


Figure 1. LO Nonlinearities for Bivariate Gaussian pdf.

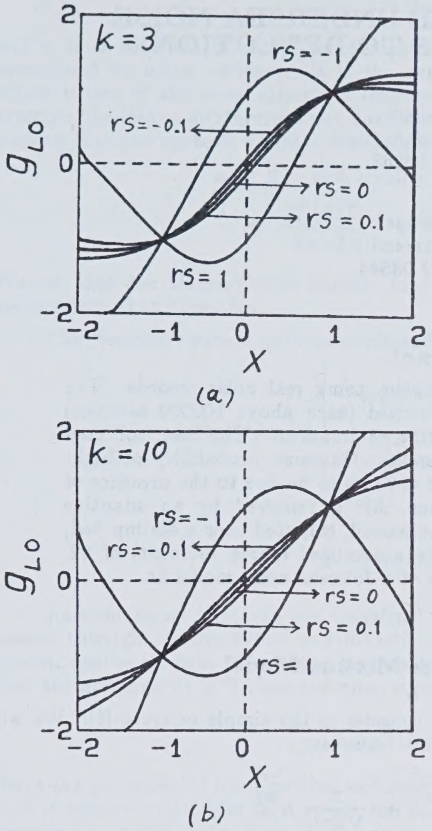


Figure 2. LO Nonlinearities for Bivariate t-distribution; (a) k=3; (b) k=10

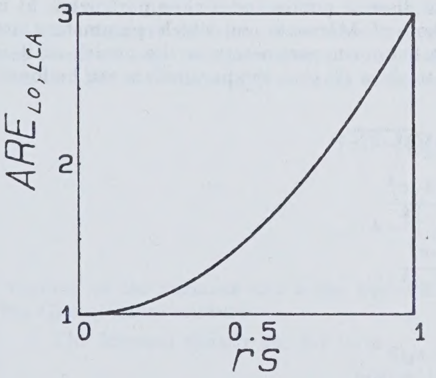


Figure 3. $ARE_{LO,LCA}$ for Bivariate Gaussian pdf.

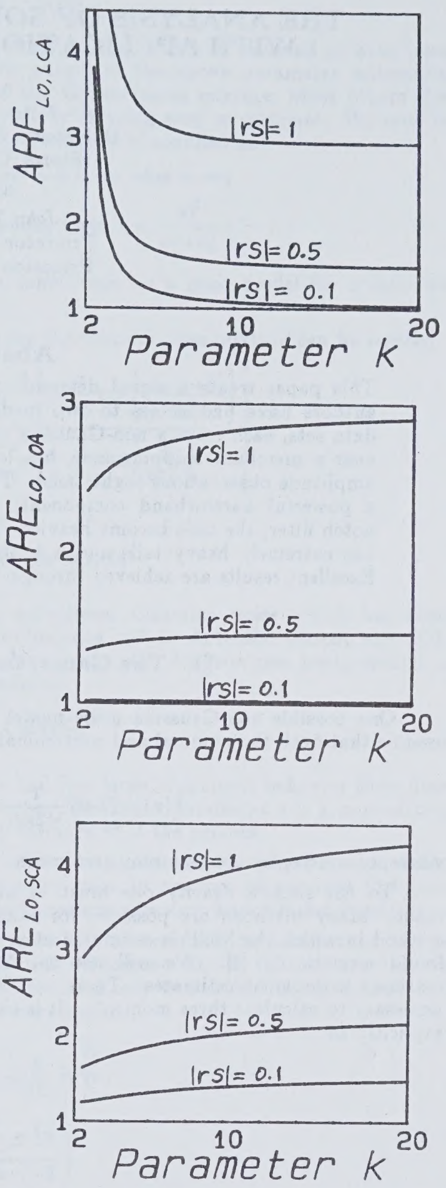


Figure 4. ARE's for Bivariate t-distribution

THE ANALYSIS OF SOME UNDERSEA NOISE WITH APPLICATIONS TO DETECTION

Peter Willett
University of Connecticut
Storrs, CT 06268.

and

John Thomas
Princeton University
Princeton NJ 08544.

Abstract

This paper treats a signal detection problem using real noise records. The authors have had access to two medium-sized (each about 100000 samples) data sets, each from a non-Gaussian marine environment. The first, collected near a merchant shipping lane, has lower-than-Gaussian probability of high-amplitude observations (light tails). This appears to be due to the presence of a powerful narrowband component; when this is removed by an adaptive notch filter, the tails become heavier. The second, collected near a shrimp bed, has extremely heavy tails and a kurtosis (normalized fourth moment) of 82. Excellent results are achieved through use of a Johnson noise model fit.

§1 The Gauss/Gauss Mixture Model

One possible non-Gaussian noise model to consider is the simple mixture [1]. We will assume that both the nominal and contaminant are Gaussian:

$$f(x) = (1-\epsilon)\frac{1}{\sqrt{2\pi}\sigma_1}\, e^{\frac{-x^2}{2\sigma_1^2}} + \epsilon\frac{1}{\sqrt{2\pi}\sigma_2}\, e^{\frac{-x^2}{2\sigma_2^2}}$$

where, of course, we are assuming zero mean.

To use such a density one must be able to estimate the parameters of the mixture model. Many methods are possible, for example the Moment Generating Function (MGF) method in which the MGF is estimated at certain discrete points and a close parametric fit is found numerically [2]. We will use the Method of Moments, in which parameters are matched to moment estimates. There are three unknown parameters in the model, so it is necessary to calculate three moments. It is easy to show [3] that the parameters can be found explicitly as:

$$\sigma_1^2 = \frac{1}{2a}[-b+\sqrt{b^2-4ac}\,]$$

$$\sigma_2^2 = \frac{e_4/3-\sigma_1^2}{1-\sigma_1^2}$$

$$\epsilon = \frac{1-\sigma_1^2}{\sigma_2^2-\sigma_1^2}$$

where

$$a = 1 - e_4/3$$

$$b = e_6/15 - e_4/3$$

$$c = (e_4/3)^2 - e_6/15$$

and e_i is an estimate of the i^{th} moment of the noise process, which is assumed to have been normalized to unity variance. It is the explicit nature of the above parameter estimation which is one of the most attractive features of the Gauss/Gauss mixture. Most others (for example the Huber mixture) do not have the property of being easy to estimate. We note in passing that the kurtosis κ achievable with a (σ_1,σ_2) mixture is bounded as

$$\kappa \geq 3 \quad \text{with equality when } \epsilon=0,1 \quad \text{or when } \sigma_1=\sigma_2$$

$$\kappa \leq \frac{(\sigma_1^2+\sigma_2^2)^2}{4\sigma_1^2\sigma_2^2} \quad \text{with equality when } \epsilon = \frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}$$

We see that the Gauss/Gauss mixture may in some cases be a good model for noise with heavier tails than Gaussian.

The Neyman-Pearson optimal nonlinearity for the Gauss/Gauss mixture can be written

$$\psi_{GG}(x) = \log\left[\frac{\frac{1}{\sigma_1}\exp\left(\frac{-(x-s)^2}{2\sigma_1^2}\right) + \frac{1}{\sigma_2}\exp\left(\frac{-(x-s)^2}{2\sigma_2^2}\right)}{\frac{1}{\sigma_1}\exp\left(\frac{-x^2}{2\sigma_1^2}\right) + \frac{1}{\sigma_2}\exp\left(\frac{-x^2}{2\sigma_2^2}\right)}\right]$$

§2 The Johnson Model

Johnson noise [4,5] can be visualized as unit-power Gaussian noise which has been passed through a memoryless and invertible nonlinearity $g^{-1}(\cdot)$. Actually, almost any i.i.d. process can be thought of as being produced in such a way, the Johnson case being special in that the nonlinearity $g^{-1}(\cdot)$ has a simple, explicit form:

$$g^{-1}(z) = \lambda \sinh\left(\frac{z}{\delta}\right)$$

Since the parameter δ has more control over the tail (i.e. large argument) behavior than does λ, it is customary to refer to it as the *tail parameter*. The other parameter λ is a normalizing constant by which, for a given δ, one can set the variance σ^2 of the process.

It can be seen that for constant σ^2 a large value of δ produces essentially Gaussian noise, while a small value causes the density to have heavy tails. In practice we see that $\delta>5$ produces approximately Gaussian densities, while $\delta<2$ characterizes very heavy-tailed densities.

As with the mixture model it is important to be able to estimate the parameters, and as with the mixture model we have developed a Method of Moments fit. In the Johnson case we see there are only two parameters to calculate, so we need only two moments, the second and fourth. We can derive [3] a moment estimate of parameters:

$$\delta = \left(\frac{2}{\log(\sqrt{2\kappa-2}-1)}\right)^{\frac{1}{2}}$$

$$\lambda = \left(\frac{2\sigma^2}{\sqrt{2\kappa-2}-2}\right)^{\frac{1}{2}}$$

where σ^2 is the variance and κ the kurtosis. This is possibly a simpler calculation than for the Gauss/Gauss mixture.

The Johnson density has the form

$$f(x) = |g'(x)|\phi(g(x))$$

where $\phi(\cdot)$ is the unit normal and g is the inverse of the sinh nonlinearity:

$$g(x) = \delta \log \left[\frac{x}{\lambda} + \sqrt{(\frac{x}{\lambda})^2+1} \right]$$

and

$$g'(x) = \frac{\delta}{\lambda \sqrt{(\frac{x}{\lambda})^2+1}}$$

The Neyman-Pearson optimal nonlinearity in the presence of an additive signal c can be written most easily as

$$\psi_J(x) = \log \left(\frac{g'(x-c)}{g'(x)} \right) - \frac{1}{2} \left([g(x-c)]^2 - [g(x)]^2 \right)$$

It can be shown that Johnson nonlinearities blank large observations; that is, in the large-amplitude limit, they are zero.

§3 Merchant Shipping Noise

This data set was collected from a single microphone suspended in the Indian Ocean near a merchant shipping lane. The sampling rate used was 1250 data points/second, and 102500 samples were taken, giving a total record length of 82 seconds.

A moment analysis of the raw data gives us the following table:

Moments of Raw Data	
mean	0.023
std dev	544.0
skewness	0.014
kurtosis	2.25

There does not seem to be any reason to assume the noise to be non-stationary.

A spectrum analysis of the noise was performed, with results shown in Figure 3.1. Instead of a broadband or low-pass characteristic, the spectrum has a strong peak around 90-100 Hz, smeared over a band of about 50 Hz. Returning to the physical system we see that this is not remarkable. This relatively high frequency tone is more likely due to vibrations in the structure of the ship, than to the rotation of the engine itself. This is also suggested by small peaks at harmonic frequencies of the heavy tone, pointing to the nonlinear nature of the device producing the tone, and suggestive of mechanical vibration.

A substantial part of the noise power is in this tone, as can be seen from the empirical spectrum, the small value of kurtosis, and light-tailed probability plot. The removal of this tone would very likely make a radical difference in the statistics of the process, since we expect to be able to remove the effect of the passing ship and deal only with ambient noise. An ideal device to accomplish this separation is the Adaptive Notch Filter.

An adaptive notch filter [6] functions by placing the deepest part of its notch over the highest amplitude section of the noise spectrum. In the common spectral-analysis test model of sinusoids in white noise, the notch will naturally place itself to reject the sinusoid. In its full generality the ANF can include many cascaded stages to reject many sinusoids. A gradient algorithm controls the adaptation and seeks to minimize output power. Considerations about convergence, and modifications to the filter which can enhance it will not be treated here; initial convergence can often be quite slow. In our case we will ignore this problem by initializing the filter to a nearly-optimal value.

The form of the filter is

$$H(z) = \frac{z^2 + bz + 1}{z^2 + \alpha b z + \alpha^2}$$

It is easy to show that this is a notch filter with

$$\text{center frequency} = \cos^{-1}(-\frac{b}{2})$$

and

$$\text{band-width} \sim 2(1-\alpha)$$

The adaptation is accomplished through updating b:

$$b_{n+1} = b_n - \mu y_n (x_{n-1} - \alpha y_{n-1})$$

where μ is a constant which trades stability for speed of convergence. A typical value for μ is $(1-\alpha)/10$.

Adaptive notch filters were applied to the raw data in an attempt to remove the noise from the ship. The effects of several notch widths on the probability distribution of the data are shown in Figure 3.2; the noise in each case has been normalized to unit variance and zero mean. As the notch became wider the distribution developed heavier tails. As the parameter α dropped below .80 very little further effect was seen on the distribution; this value of α corresponds to a notch width of 80 Hz and is of the same order but slightly larger than the width of the smeared peak in Figure 3.1. That $\alpha < .80$ corresponds to an unnecessarily sloppy notch is confirmed in Figure 3.3, a plot of output power versus α. Qualitatively we observe that for $\alpha > .95$ the filter removes too little of the ship's noise; the improvements are unimpressive for $\alpha < .80$, and in a practical situation such a filter could remove too much signal power to be worthwhile.

This work suggests the following model for merchant shipping noise X_n:

$$X_n = Y_n + Z_n$$

where Y_n is a high power, narrowband process centered around 94 Hz., and probably due to mechanical vibration in the nearby ship; and Z_n is a lower power, more broad-band process with somewhat heavier-than-Gaussian tails, possibly unrelated to the ship. An empirical probability plot of the output of the ANF with $\alpha = .90$ is shown in Figure 3.4.

The application of these studies and models is to be able to detect a known signal $\{c_i\}$ in noise. Several detectors have been implemented.

[D1] The Linear Detector.

[D2] The Gauss/Gauss Mixture Detector.
The parameters:

$$\epsilon = 0.0371 \qquad \sigma_1 = 483 \qquad \sigma_2 = 1050$$

are the same as in [2].

[D3] The amplifier-limiter.
This detector limits input amplitudes to $\pm 2\sigma$, and was included after [2].

[D4] The Notch-Linear Detector.
This detector is simply D1 where observations have been pre-processed by a notch filter with $\alpha = .90$. Owing to the interference of signal with adaptation, a non-adaptive notch filter with center frequency 94 Hz was used. In practice, this scheme would have to be used with the signal random, or frequency-hopped, or weak, or some scheme to avoid having the ANF cancel the signal. There is not enough data present to test such a procedure adequately in the present situation.

The simulations were performed as follows. The noise power was estimated and used to obtain a sample-by-sample SNR of $0dB$. A constant signal was used, after [2], and also to avoid a situation where the signal spectrum was too optimistically or too pessimistically placed with respect to the noise spectrum (for example, a non-vanishing alternating signal could be detected with almost unity power). The record length was chosen as 10, allowing approximately 10000 repetitions per detector. Note that observations were made down to $\alpha = 10^{-4}$, but only those where $\alpha > 10^{-3}$ were considered reliable. (The notation is standard but regrettable: now that we are not dealing with notch filters, α is used to mean probability of false-alarm.)

The results were as shown in Figure 3.5. The linear detector (D1) outperformed both the mixture detector (D2) and the amplifier-limiter. This is not surprising considering that the latter two are optimal for noise with heavier-than-Gaussian tails, and this data set has lighter-than-Gaussian tails. Pre-processing the data by a notch-filter offered a dramatic improvement over the first three detectors. A further simulation, shown in Figure 3.6, demonstrates the effect of widening the notch on detector performance.

As suggested earlier by Figure 3.4, the output from the notch filter has slightly heavier-than-Gaussian tails, and a parametric fit can be achieved. A further simulation was performed to see if there were any significant improvement from the increased sophistication of the model. This included detectors:

[D5] The Notch-Mixture Detector.
The output of the notch filter was passed through a mixture nonlinearity with parameters

$$\epsilon = .33 \qquad \sigma_1 = 337 \qquad \sigma_2 = 401$$

[D6] The Notch-Johnson Detector.
The output of the notch filter was passed through a Johnson nonlinearity with parameters

$$\delta = 6.79 \qquad \lambda = 2410$$

The results indicate that very little improvement is achieved [3].

§4 Shrimp Noise

This data set was collected in the Pacific Ocean off Hawaii, near a concentration of shrimp. The sampling rate used was 40000 data points/second, and 96000 samples were taken, giving a total record length of 2.4 seconds.

A moment analysis of the raw data gives us

Moments of Raw Data	
mean	1.79
std dev	48.3
skewness	0.089
kurtosis	82.5

This is characteristic of a highly non-Gaussian environment; the density has extremely heavy tails. A probability plot drawn in Figure 4.1 confirms this: approximately one out of ten thousand observations will have amplitude greater than 16σ.

In this figure the Gauss/Gauss mixture and Johnson fits are also presented. The relevant parameters are for the mixture

$$\epsilon = .00165 \qquad \sigma_1 = 43.4 \qquad \sigma_2 = 563$$

and for the Johnson system

$$\delta = .901 \qquad \lambda = 20.8$$

It is apparent that the mixture fit is not acceptable. For such heavy-tailed noise, the probability plot is forced to develop a 'knee' (around 3.5σ) which is not present at all in the empirical distribution. The Johnson fit on the other hand is quite good, at least up to a tail weight of 10^{-4}. Beyond that, shrimp noise is heavier-tailed than the Johnson fit, although the fit does not seem to be degrading quite as rapidly as for the Gauss/Gauss mixture. It is clear that the Johnson density is a good sub-optimal parametric choice.

The physical system producing this noise sequence is a nearby school of shrimp. To communicate with each other (and presumably to attract a mate) these creatures snap their claws rapidly, producing loud 'clicks'. A noise sequence from sample 10001–20000 is shown in Figure 4.2a. In this plot there are easily-recognizable high-amplitude events, of about two to five milliseconds duration, occurring against a more well-behaved background. Unfortunately, although in this record the clicks are loud and of reasonable length, the next record (samples 20001–30000, Figure 4.2b) demonstrates that this is not always the case. There are several overt clicks, and also many high-amplitude observations which seem to be isolated. The reason for these isolated observations is not clear; we can speculate that the longer-duration clicks are produced differently by the shrimp themselves. We also note a tendency (which is unfortunately not a rule) for two events to occur within a space of 10–30 ms or so, perhaps a result of both claws being exercised.

The difficulty in characterizing high-amplitude observations into well-behaved bursts makes makes recognition of mode-changes complicated. It was decided, due to the heavy-tailed nature of the noise, to attempt to separate clicks from background noise using the non-parametric burst detector [7]. Such a device functions by deciding on a sample-by-sample basis (i.e. by a simple threshold) from which mode a data point comes, and subsequently smoothing this estimate using a median filter. Figure 4.3 gives close-up views of fairly typical well-defined clicks. It was decided to use a median filter of length 61 since that is the approximate length of the shorter clicks. The short median filter necessary to detect such quick bursts can lead to too many spurious mode-changes, so a fairly high threshold of twice the standard deviation was used.

In order to compensate for the varying burst-strength (probably due to the proximity of the source shrimp), each burst in the "high" mode has been normalized to have the same variance. This empirical approach to mode separation has had some success: the tails on both the high- and low-variance modes have been reduced. This can be seen from the separated moment and Johnson parameters

Moment and Johnson Parameters

	Original	Analysis After Separation		
		Low	High	Raw High
std dev	48.3	35.1	-	110.0
kurtosis	82.5	4.47	4.90	28.2
δ	0.90	2.01	1.85	1.03
λ	20.8	62.3	-	67.1

The "high" mode data does not of course contain any relevant amplitude-related parameters. The "raw high" column refers to non-normalized bursts and shows us that a simple (i.e. multiplicative) burst detector is not optimal in this situation.

The non-stationary nature of shrimp noise makes observations about time dependence difficult and perhaps not very meaningful; however, we show the spectrum in Figure 4.4. There is a very sharp rise at 2.5 kHz and a steady drop-off, with some evidence of a harmonic tone around 5 kHz. The plot's lack of smoothness, despite the application of a Parzen window, attest to the unmanageability of this data set. It also appears that sampling was done at something below the Nyquist frequency, although it is not clear that a reasonable one exists for this environment. For the earlier Figure 4.3, it was observed that a burst seemed somewhat tonal (around 2.5 kHz): to examine this hypothesis the high- and low-variance modes were treated separately. In each case, data records of length 64 (this low length was chosen due to the difficulty of finding long bursts) were transformed, squared, and averaged; the results are shown in Figure 4.5. The bursts appear considerably more tonal and band-limited in nature than the background mode, and indeed the small 2.5 kHz peak in the low-variance mode may be due to imperfections in the mode-separation algorithm.

The simple detection described in the Merchant Shipping Noise section was posed in this noise environment to compare several detection schemes. These were:

[D1] The Linear Detector.

[D2] The Gauss/Gauss Mixture Detector.

[D3] The Laplace Noise Detector.

[D4] The Johnson Noise Detector.

[D5] The Mode-Switched Johnson Detector.
Ideally to use this detector the two modes should, after removal of a multiplicative constant, have identical densities. Since this is not the case here, the more pessimistic 'raw high mode' values of

$$\delta=1.03 \qquad \lambda=67.1$$

were used here. The non-parametric mode-detector described earlier controlled premultiplication: observations judged to have come from the high-variance mode were multiplied by one third, the approximate ratio of the standard deviations of the two modes.

In each case a sample-by-sample signal to noise ratio of 0 dB was used, a constant signal, and a detection record length of 5. The results of this experiment were rather as expected. As shown in Figure 4.6, the linear detector (D1) performed extremely poorly. The Johnson detector performed well, better than the Gauss/Gauss mixture detector (D2) or the Laplace detector (D3) (although this was quite good) as was expected from its good distributional fit. The mode-switched Johnson detector (D5) was the best simulated although due to burst-estimation problems the performance deteriorated for small probabilities of false-alarm. It is a somewhat unsatisfying system to use, however, due to the necessity to specify parameters of burst length and threshold, and the difficulty of burst-detection in the presence of a non-vanishing signal. It would seem that the Johnson detector is perhaps the wisest choice in this environment.

§5 Summary

The analysis of the previous data sets has suggested, among others, the following insights.

The Johnson detector seems dependably good, and is particularly attractive since its two parameters can be estimated explicitly through the variance and kurtosis statistics (this of course assumes zero-mean, symmetric densities).

The application of a notch filter to the data stream seems often to be an easy way to improve performance, since two of the three records studied were dominated by a single tone. However, we note that care must be taken in the case of an *adaptive* notch filter, due to the unavoidable tendency of this device to cancel the received signal itself, given a narrowband signal with sufficient power. Random signals, frequency-hopped or swept signals, or a noise-alone channel would give practical inputs for such a system. The simulations presented were performed using a fixed notch filter.

The use of burst detectors seems to improve performance dramatically. However, these devices are difficult to implement in the presence of a non-vanishing signal; again, in the simulations the modes were estimated offline. Also, it was difficult to make such a system error-free, causing the performance to deteriorate to no better (or worse) than memoryless systems for low probabilities of error.

There are clearly many avenues for further research. The good performance of the Johnson detector suggests that it is not vital to have the precisely optimal density estimate in order to have a good detector. It seems possible that the complex Johnson nonlinearity could be simplified through use of some piecewise fit with little degradation in performance. Also, some study of the performance of the adaptive notch filter in the presence of a non-vanishing signal should be undertaken.

It would be valuable to improve the performance of the switched-burst detector by recognizing the imperfect nature of the switch-estimator; the resulting detector would be forced to develop a more robust low-variance-mode nonlinearity. Finally, it seems generally true that bursts are seldom white, but are more frequently narrowband. Switch-estimators, on the other hand, tend to assume bursts of white noise. One approach to estimation of highly-correlated bursts is to decimate; it would seem that a more rigorous statistical calculation might lead to a different approach with superior performance.

§6 Acknowledgment

This work was supported in part by the Office of Naval Research under Grant N00014-81-K-0146 and in part by the National Science Foundation under Grant ECS83-17777.

§7 References

[1] J.H. Miller and J.B. Thomas, "The Detection of Signals in Impulsive Noise Modelled as a Mixture Process" *IEEE Transactions on Communications,* Vol COM-24, pp. 559-563, May 1976.

[2] M. Bouvet and S.C. Schwartz,"Underwater Noises: Statistical Modelling, Detection, and Normalization", *submitted to Journal of the Acoustical Society of America,* 1986.

[3] P. Willett,"Density Representations with Applications to Signal Detection", *PhD Thesis, Princeton University,* 1986.

[4] E. Modugno,"The Detection of Signals in Impulsive Noise", *PhD Thesis, Princeton University,* 1982.

[5] N. Johnson, "Systems of Frequency Curves Generated by Methods of Translation", *Biometrika,* Vol 36, pp. 149-176, 1949.

[6] S. Czarnecki, "Nearly-Optimal Detection of Signals in Non-Gaussian Noise", *PhD Thesis Princeton University, 1983.*

[7] L. Pearlstein and B. Liu, "Retrieval of Sinusoidal Signals by Adaptive Notch Filtering", *Proceedings of the Twenty-Third Annual Allerton Conference on Communications, Control, and Computing,* pp. 574-583, 1985.

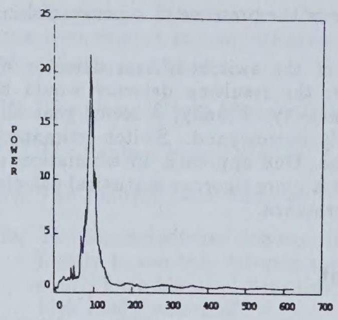


Fig. 3.1 (Merchant) Spectrum

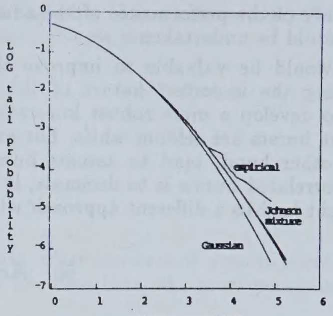


Fig 3.4 (Merchant) PDF with α=.90

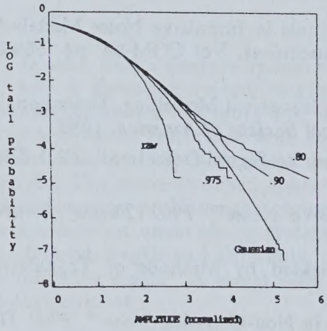


Fig. 3.2 (Merchant) PDF vs. α

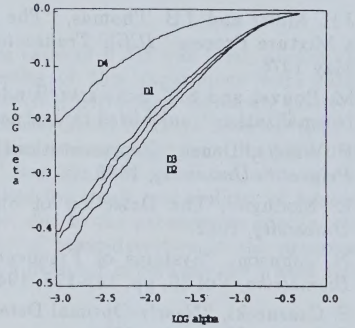


Fig 3.5 (Merchant) ROC

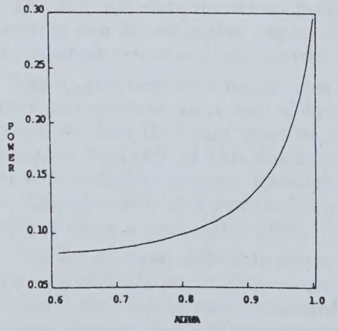


Fig. 3.3 (Merchant) Power vs. α

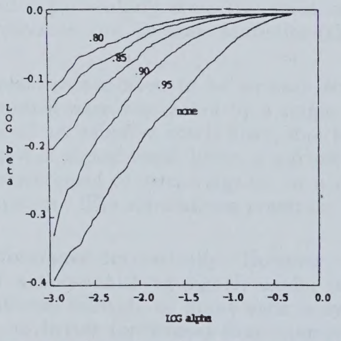


Fig 3.6 (Merchant) ROC vs. α of ANF

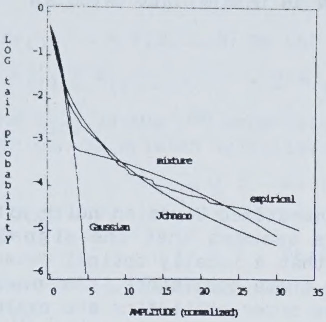


Fig 4.1 (Shrimp) PDF

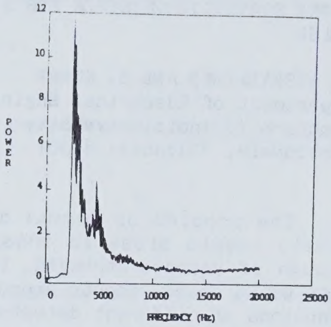


Fig 4.4 (Shrimp) Spectrum

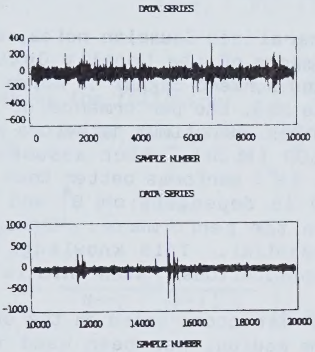


Fig 4.2 (Shrimp) Data Samples

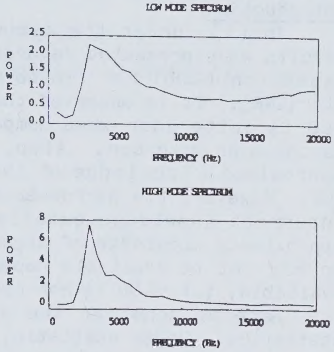


Fig 4.5 (Shrimp) Spectrum after Mode-Separation

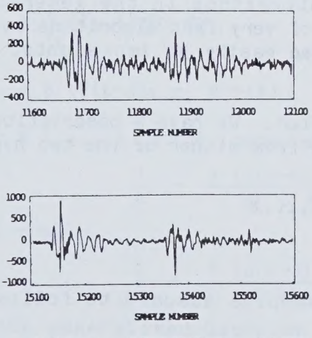


Fig 4.3 (Shrimp) Well-Defined "Clicks"

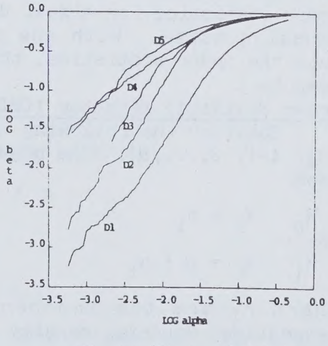


Fig 4.6 (Shrimp) ROC

ORDER STATISTIC DETECTOR FOR SIGNAL DETECTION IN GENERALIZED GAUSSIAN NOISE

R. VISWANATHAN AND B. KUMAR
Department of Electrical Engineering
Southern Illinois University
Carbondale, Illinois 62901

ABSTRACT

The problem of signal detection in generalized Gaussian noise with finite sample sizes is considered. It is assumed that the signal, though of unknown strength, is not so weak that a locally optimal detection would be considered appropriate. Under these conditions, the power functions of different detectors based on the order statistics are evaluated. A definition of Cumulative Power Loss (CPL) is formulated in order to assess the performance of these detectors. Among all the order statistics detectors, the order statistics detectors, close to the median, perform the best in the sense of minimum CPL.

Introduction

In [1], under the assumption of generalized Gaussian noise, some results were presented regarding the performance of the Locally Optimal Detection (LOD) for finite sample sizes and nonzero Signal to Noise Ratio (SNR). It is observed that for moderate SNR, the performance of the LOD is quite poor when compared to some other suboptimum detectors such as the sign detector. Also, a modified LOD (MLOD), which assumes an approximate knowledge of the signal level (Θ^*) performs better than the LOD. However, the performance of the MLOD is dependent on Θ^* and any incorrect knowledge entails some loss in the performance. Hence, an approximate knowledge of signal level is essential. This knowledge may or may not be available depending on the applications. When this is not available, the MLOD is not useful.

Here we consider the performance of detectors based on the order statistics. Order statistic, especially the median, has been used very commonly in filtering applications involving signals in heavy tailed noise. Bovik, Huang, and Munson [2] have discussed the use of median in filtering the image signals. They found the median to be very effective in suppressing the additive heavy tailed noise at least for nearly a constant signal. We investigate the usefulness of the median and other order statistics for signal detection applications in the generalized Gaussian noise. With the availability of very fast algorithms to compute the order statistics, this detector can easily be implemented on a computer.

Order Statistic Detector (OSD)

Consider the following detection problem. We have N observations $\{X_i: i=1, 2, \ldots, N\}$. The observations come from either of the two hypotheses:

$$H_0 \quad X_i = n_i \qquad\qquad i = 1,2,\ldots N$$

$$H_1 \quad X_i = \Theta + n_i$$

where n_i are the independent noise samples assumed to follow the generalized Gaussian density given by

$$f(x) = \frac{c\ \alpha^{1/c}\ e^{-\alpha|x|^C}}{2\ \Gamma\ (1/c)} \tag{1}$$

$c > 0$, and $0 < \alpha < 1$.

Let us denote the CDF corresponding to the above density by $F(x)$. Let $\{X_{(i)}: \; i = 1,2,..,N\}$ be the ordered set i.e.

$$X_{(1)} \leq X_{(2)} \leq \cdot \cdot \cdot \cdot \cdot \leq X_{(N)} \tag{2}$$

and $X_{(r)}$ be the r^{th} order statistic. Then the r^{th} order statistic detector can be written as follows:

$$T(\underline{X}) = X_{(r)} \begin{cases} \geq t & \text{Decide } H_1 \\ < t & \text{Decide } H_0 \end{cases} \tag{3}$$

where t is the threshold chosen to get a desired false alarm probability (P_{FA}).

If $\{X_i\}$ are distributed as $F(x-\theta)$, the order statistic $X_{(r)}$ is distributed with the following pdf [4].

$$f_r\,(X-\theta) = \frac{N}{(N-r)\,!\,((r-1)\,!}\,F^{r-1}(X-\theta)\,(1-F(X-\theta))^{N-r}_{*} \tag{4}$$

$$f(X-\theta)$$

The false alarm probability (P_{FA}) and the detection probability (P_D) are given by the following:

$$P_{FA} = \Pr\,\{X_{(r)} \geq t \mid \theta = 0\}$$

$$= \int_t^{\infty} f_r\,(x)\,dx$$

$$= \frac{N!}{(N-r)\,!\,(r-1)\,!} \int_{F(t)}^{1} F^{r-1}\,(1-F)^{N-r}\,dF \tag{5}$$

$$= \beta\,(F(t);\; r;\; N-r+1)$$

$$P_D = P_r\,\{X_{(r)} \geq t \mid \theta > 0\}$$

$$= \int_t^{\infty} f_r\,(x-\theta)\,dx \tag{6}$$

$$= \beta\,(F(t-\theta);\; r;\; N-r+1)$$

where $\beta\,(.;.;.)$ is the incomplete beta function. Also,

$$F(t-\theta) = \begin{cases} \dfrac{1}{2} - \dfrac{\gamma\,(\alpha(\theta-t)^c;\; 1/c)}{2\Gamma\,(1/c)} & \theta \geq t \\[2ex] \dfrac{1}{2} + \dfrac{\gamma\,(\alpha(t-\theta)^c;\; 1/c)}{2\Gamma\,(1/c)} & \theta < t \end{cases} \tag{7}$$

for the generalized Gaussian density. For the purpose of computation, (5) and (6) can be expressed as

$$P_{FA} = \binom{N}{r} \sum_{K=0}^{N-r} (-1)^K \binom{N-r}{K} \frac{r}{(r+K)} \,(1-F(t)^{r+K}) \tag{8}$$

$$P_D = \binom{N}{r} \sum_{K=0}^{N-r} (-1)^K \binom{N-r}{K} \frac{r}{(r+K)} (1-F(t-\theta)^{r+K}) \tag{9}$$

(8) is used to calculate t for a particular P_{FA}. Then P_D is computed as a function of Θ.

The Figs. (1-3) show the power curves for the order statistic detectors for the generalized Gaussian noise with $P_{FA} = 0.1$ and N=8.

Fig. (2) shows the power curves for r = 2,3,...,7 and c = 0.5. The SNR range is between -24 dB and -4 dB. $X_{(3)}$ gives the best probability of detection below -20 dB. The median, $X_{(4)}$, performs best for SNR between -20 dB and -10 dB. Similar behavior of the power curves, though over different SNR ranges, is observed for c = 0.25 and c = 1.0. We see that the best order statistics detector performs significantly better than the LOD, MLOD, SD, and SSD in the SNR range considered. Also, the OSDs are consistent i.e. power function approaches 1 as $\theta \to \infty$.

Average Performance of OSD

We have evaluated the average performance of the OSD by introducing the concept of cumulative power loss (CPL). The cumulative power loss is defined as the probability of miss summed over the entire signal range, for a particular P_{FA} i.e.

$$\mathrm{CPL}\,(P_{FA}) \triangleq \int_0^\infty [1 - P_D\,(\theta,\, P_{FA})]\, d\theta \tag{10}$$

Existence of CPL:

From (9) and (10) we have

$$\mathrm{CPL} = \sum_{K=0}^{N-r} (-1)^K \binom{N-r}{K} \binom{N}{r} \left(\frac{r}{r+K}\right) \int_0^\infty F^{r+K}(t-\theta)\, d\theta \tag{10a}$$

When the integral is bounded, the CPL exists. Since $F^{r+K}(\ )$ is the CDF of the $(r+K)^{th}$ order statistic from a sample of size r+K, the existence of the integral is guaranteed if the expected value of the $(r+K)^{th}$ order statistic exists. The existence of the expected value of the Generalized Gaussian noise guarantees this and hence the CPL is bounded.

The detector with the minimum CPL would give the best average performance. This has been computed numerically for c = 0.5 and 1.0 and P_{FA} = 0.001, 0.01, and 0.1. The results are presented in Table 1 below.

TABLE 1

Cumulative Power Loss

N=8

↓r	CPL →					
	C = 0.5			C = 1.0		
	$P_{FA} = 0.1$	$P_{FA} = 0.01$	$P_{FA} = 0.001$	$P_{FA} = 0.1$	$P_{FA} = 0.01$	$P_{FA} = 0.001$
1	1.07	1.285	1.362	0.961	1.338	1.557
2	0.457	0.558	0.696	0.59	0.865	1.129
3	0.243	0.401	0.613	0.436	0.746	1.051
4	0.189	0.441	0.759	0.393	0.768	1.133
5	0.239	0.626	1.092	0.43	0.892	1.319
6	0.368	0.997	1.755	0.529	1.126	1.696
7	0.658	1.815	3.26	0.719	1.567	1.921
8	1.604	4.848	9.24	1.216	2.807	4.419

TABLE 2

Cumulative Power Loss

N=11

↓r	CPL →					
	C = 0.5			C = 1.0		
	P_{FA} = 0.1	P_{FA} = 0.01	P_{FA} = 0.001	P_{FA} = 0.1	P_{FA} = 0.01	P_{FA} = 0.001
1	1.186	1.46	1.667	0.976	1.389	1.734
2	0.536	0.672	0.768	0.614	0.905	1.033
3	0.323	0.407	0.442	0.465	0.695	0.85
4	0.203	0.33	0.385	0.372	0.605	0.774
5	0.262	0.439	0.596	0.369	0.648	0.753
6	0.156	0.38	0.598	0.336	0.655	1.488
7	0.21	1.523	0.794	0.375	0.78	1.179
8	1.295	0.763	2.88	0.441	0.937	1.311
9	0.448	1.185	2.398	0.547	1.152	2.443
10	0.753	1.989	3.128	0.737	1.628	2.393
11	1.749	5.173	10.138	1.23	2.762	4.464

Table 1 shows the CPL for $P_{FA} = 0.1$. The minimum value of CPL occurs at $r = 4$ for $c = 0.5$ as well as $c = 1.0$. However, for the lower false alarm probabilities $X_{(3)}$ has lower CPL than $X_{(4)}$, hence $r = 3$ is a better choice as an OSD. But since the difference in CPL is not very significant, $X_{(4)}$ could also be used without any significant degradation in the performance. Results for sample size equal to 11 are shown in Table 2.

Trimmed Mean Detector (TMD)

In the previous section, the order statistics were compared for the detection of signal in the presence of the generalized Gaussian noise. The median of the sample performed rather well for the moderate SNRs. Now we consider a more general case of the median, where we discard the K smallest and the K largest observations and use the mean of the remaining observations as the test statistic. This test statistic is known as the trimmed mean [3]. If $K/N = \nu < 1/2$, we shall denote the resulting trimmed mean as $\overline{X}_{\nu}$ which is given by

$$\overline{X}_{\nu} = \frac{1}{N - 2K} \{X_{(K+1)} + \cdot \cdot \quad + X_{(N-K)}\} \tag{11}$$

The mean is the extreme case of the trimmed mean for $\nu = 0$.

We have looked at the power curves for the simulated TMDs for the generalized Gaussian noise. Figs. 4-6 show the power curves of TMDs for $c = 0.5$, 1.0, and 1.5, $\nu = 0.125$, and .250, for the sample size of eight.

For $c = 0.5$ and 1.0, when the noise is extremely heavy tailed, $\overline{X}_{.25}$ performs better than $\overline{X}_{.125}$ as we would expect. However, the median ($X_{(4)}$) (Figs. 2 and 3) is better than both $\overline{X}_{.25}$ and $\overline{X}_{.125}$. When $c = 1.5$, the tails are only moderately heavy, therefore we can see that $\overline{X}_{.25}$ competes very closely with $X_{(4)}$ in the moderate SNR range.

CONCLUSION

The performances of different detectors based on the order statistics are considered for signal detection in generalized Gaussian noise. A performance index, termed cumulative power loss, is defined. Among the order statistics detectors, the order statistics detectors, close to the median, perform the best in the sense of minimum CPL.

REFERENCES

[1] B. Kumar and R. Viswanathan, "Moderate Signal to Noise Ratio and Locally Optimal Detection in Generalized Gaussian Noise," to appear in the Proceedings of XX Annual Conference on Information Sciences and Systems, Princeton, NJ, March 19-21, 1986.

[2] A. C. Bovik, T. S. Huang and D. G. Munson Jr., "A Generalization of Median Filtering Using Linear Combinations of Order Statistics," IEEE Trans. on Acoustics, Speech and Signal Processing, Vol. ASSP 31, No. 6, December 1983.

[3] E. L. Lehman, "Theory of Point Estimation," John Wiley and Sons, New York, pp. 360-368, 1983.

[4] Jean D. Gibbons, "Nonparametric Statistical Inference, Marcel Dekker, Inc., New York, Second Edition, 1985.

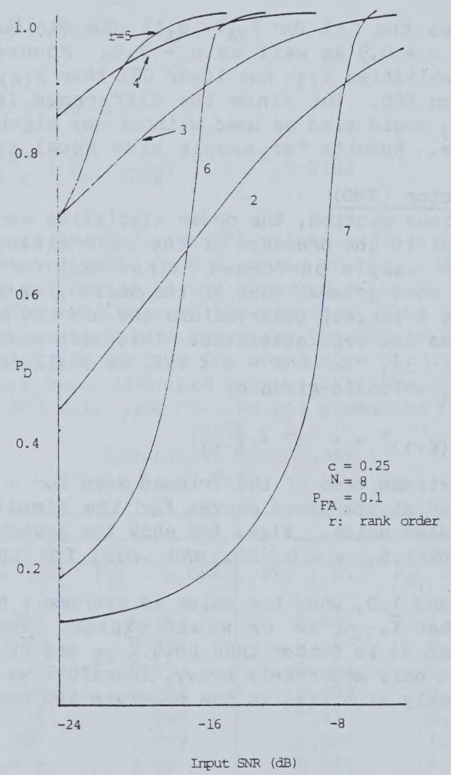


Fig. 1 Moderate Signal Performance of Order Statistics Detectors.

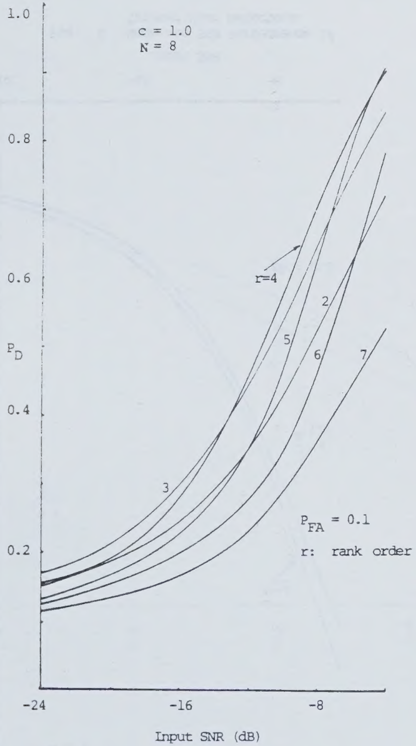


Fig. 3 Moderate SNR Performance of Order Statistics Detectors.

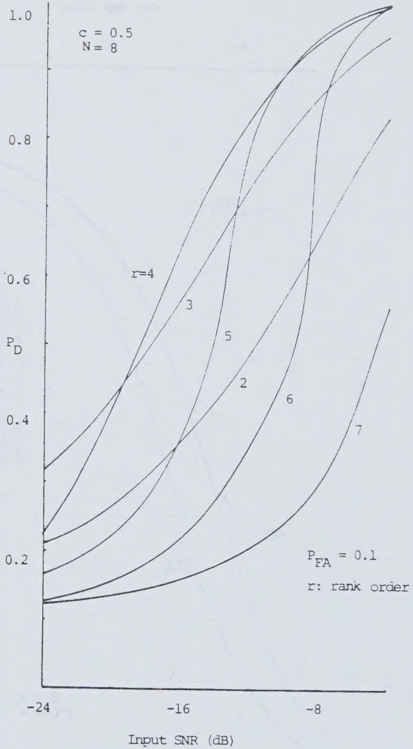


Fig. 2 Moderate SNR Performance of Order Statistic Detectors.

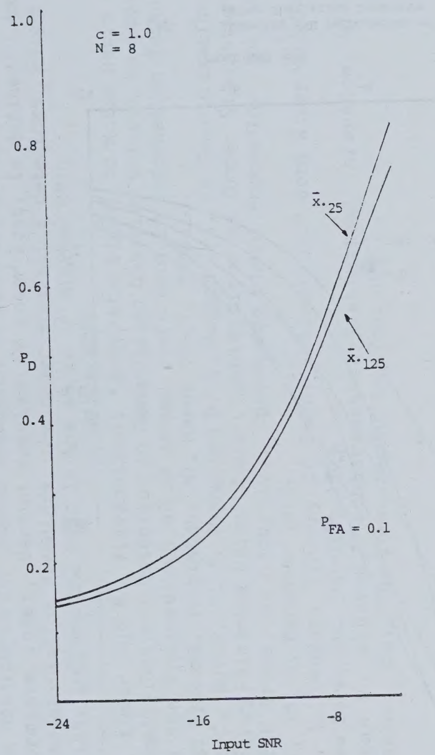


Fig. 5 Moderate SNR Performance of Trimmed Mean Detectors.

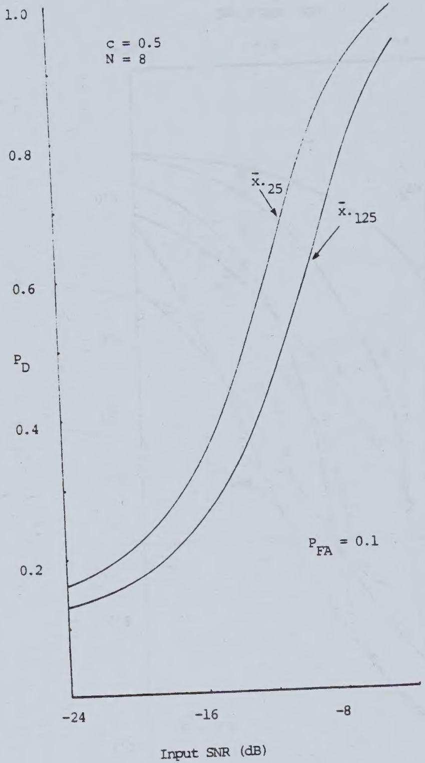


Fig. 4 Moderate SNR Performance of Trimmed Mean Detectors

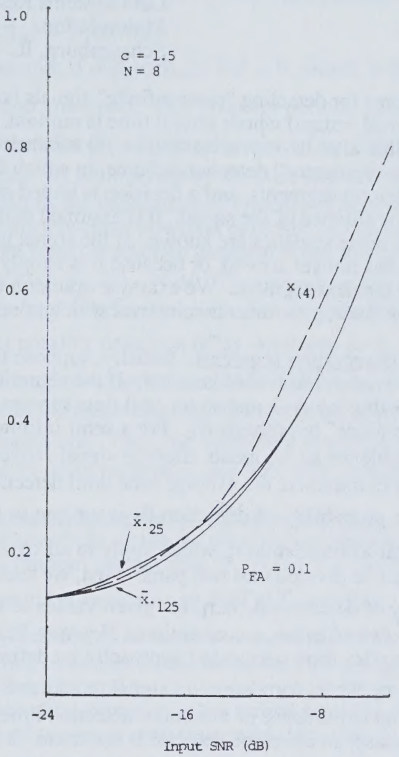


Fig. 6 Moderate SNR performance of Trimmed Mean Detectors.

TIME-SEGMENTED DETECTION OF A RANDOMLY ARRIVING SEMI-INFINITE SIGNAL

L.A. Longley and C.C. Lee
Dept. of Electrical Engineering
and Computer Science
Northwestern University
Evanston, IL 60201

L.A. Longley
also with
Data Systems Research Lab
Motorola Inc.
Schaumburg, IL 60196

This paper examines procedures for detecting "semi-infinite" signals (signals which are present for all time after their arrival instant) whose arrival time is random. It is desired to detect the signal as soon as possible after its arrival instant, i.e., to minimize the detection delay. We propose using a "time-segmented" detection scheme, in which the observed signal is divided into successive fixed-length segments, and a decision is issued at the end of each segment regarding the presence or absence of the signal. It is assumed that the signal has a "constant envelope", and that the noise statistics are known. If the signal is not detected in a given segment, either because it has not yet arrived, or because it is simply "missed", then the detection process continues with the next segment. We examine optimum detector structures and an efficient search scheme for finding the detection interval which minimizes the detection delay.

A. Arrivals at beginnings of observation segments. Initially, suppose that the signal can arrive only at the start of an observation interval of length T. If the signal is missed, a detection attempt will be made in the following interval, and so on. All time segments before the signal arrives are described by a "noise-alone" hypothesis H_0. For a semi-infinite signal, a "signal-present" hypothesis H_1 is applicable for all segments after the signal arrives, beginning with the arrival segment. It is desired to minimize the average time until detection, which is shown to be $E_1(T_D) = T / \beta_1(\alpha,\eta)$. The probability of detection β_1 is written in terms of the false-alarm probability α and the signal-to-noise ratio η, which apply to all signal-present segments.

The optimization problem can be divided into two parts. First, we seek a detector structure which maximizes the probability of detection $\beta_1(\alpha,\eta)$ for given values of α and η in the observation segment. For the above criterion, a conventional Neyman-Pearson detector is optimum (given that we are using this time-segmented approach): by definition, such a detector maximizes β_1, given α, for any η. Thus, for a constant signal in additive white gaussian noise, a linear detector is optimum in the sense of minimum detection time; similarly, for a sinusoidal signal with random phase, an envelope detector is optimum. It is assumed that the test threshold is adjusted to achieve the specified value of α, based on knowledge of the noise environment.

Second, we may adjust T to globally minimize $E_1(T_D)$. If the segment length is too short, β_1 may be so low that detection never occurs; on the other hand, if T is too long, then detection may be assured, but the average delay will be excessive (note that since β_1 has a maximum value of unity, $E_1(T_D)$ has a minimum value of T). For a detector continuously searching for a randomly arriving signal, a "false-alarm frequency" $f_\alpha = \alpha / T$, the average number of false alarms per unit time, may be used to indicate how often the system is falsely informed of the presence of the signal being searched for, in a manner which is independent of the segment length. In terms of this "normalized time" representation, the above expression for the mean time until detection may be rewritten in "normalized" form: $E_1(T_D) \cdot f_\alpha = \alpha / \beta_1(\alpha,\eta)$. Then, the solution to minimizing mean time until detection involves minimization, with respect to α, of the right-hand side of this equation. Once the normalized solution is found, it may be de-normalized for any particular choice of f_α.

Now, minimizing α / β_1 with respect to α requires an explicit statement of the (assumed) dependence of η on α: $\eta = (T/T_\alpha) \cdot \eta_\alpha = \alpha \cdot \eta_\alpha$, where η_α is the ratio of the signal energy in a "false-alarm interval" $T_\alpha \equiv 1/f_\alpha$ to the noise power spectral density. In other words, we assume that the signal energy per unit time is constant, so that the signal energy in an observation period depends linearly on the length of that period. Then, for a prespecified value of η_α, the detection probability may be rewritten as a function of the single parameter α:

$\beta_1(\alpha,\eta) = \beta_1(\alpha,\alpha\eta_\alpha) \equiv \beta_1(\alpha;\eta_\alpha)$

It may be shown that if $\partial^2\beta_1(\alpha;\eta_\alpha) / \partial\alpha^2 < 0$, which is frequently the case, then $E_1(T_D)\cdot f_\alpha$ has a unique minimum. This property permits us to employ an efficient "binary" search scheme to determine the minimizing value of α. Finally, the minimum $E_1(T_D)\cdot f_\alpha$ may then be "denormalized" by multiplying by the false-alarm interval T_α. Use of a numerical scheme is necessitated by the complexity of typical cumulative detection probabilities, even for relatively simple uniformly-most-powerful detectors. Consider, for example, the detection of a sinusoid with random phase in additive gaussian noise, using a conventional envelope detector (ED) statistic: $L_E \equiv |({}^1/_T) \int^T{}_0\, r(t)\, e^{j\omega t}\, dt|$. The corresponding probability of detection is well known to be described by a Rician function. Numerical results indicate that, as would be expected, the lowest possible detection delay decreases as η_α increases. In other words, for a given η_α, the best choice of α occurs as a tradeoff between its ability to decrease $E_1(T_D)\cdot f_\alpha$ through its presence in the numerator of that expression, and a corresponding decrease of $\beta_1(\alpha;\eta_\alpha)$ in the denominator. Thus, the increasing detection probability (with η_α) results from the increasing signal-to-noise parameter, and occurs in spite of the decreasing false-alarm probability.

B. Arrivals distributed throughout observation segments. It is more realistic to assume that the signal can arrive anywhere within the observation interval. For the least favorable arrival distribution for the random arrival time T_1, $p_a(t_1) = 1 / T$, $1 \le t_1 \le T$, the (normalized) mean time until detection evaluates to $E_1(T_D)\cdot f_\alpha = \alpha\, [{}^1/_2 + (1-\beta_o) / \beta_1]$, where β_o is the average probability of detection in the arrival segment, and β_1 still is used to describe the probability of detection in all blocks following the arrival segment, in case a "miss" occurs in that first segment. The first term is the mandatory half-segment wait (on average) which occurs even with successful detection in the arrival segment, while the second term represents the average number of segments until successful detection. For high signal-to-noise ratios, both β_o and β_1 approach unity, so that the second term vanishes.

It is apparent that maximizing β_o and β_1 will minimize $E_1(T_D)$, for a given segment length T. However, the detector structure which maximizes β_o will not, in general, maximize β_1, and vice versa. In fact, designing a detector structure to minimize $(1-\beta_o) / \beta_1$ for a given segment length is extremely difficult. Instead, we examine the performance of two detectors for sinusoids with random phase, the ED and an "Envelope Ramp Detector" (ERD), which includes a "ramp" weighting factor giving increasing significance to the waveform later in the observation segment: $L_R \equiv |({}^{\sqrt{3}}/_T) \int^T{}_0\, r(t)\, e^{j\omega t}\, ({}^t/_T)\, dt|$. The ERD is specifically designed for detecting a sinusoid with uniform arrival distribution, approximately maximizing the probability of detection in the arrival segment (β_o).

A numerical comparison of these detectors assuming a uniformly distributed arrival time indicates that the ED outperforms the ERD, throughout the range of signal-to noise ratios examined. This somewhat surprising result may be rationalized as follows: while β_o for the ERD always exceeds that for the ED, β_1 for the ERD is surpassed by that for the ED. Evidently, for optimum values of α, β_o is low enough that the chance of a "miss" is significant; when a miss indeed occurs, the inferior subsequent performance of the ERD has a "dominant" effect.

A Comparison of Error Probability Computations using Large Deviations Methods for Detectors with Dependent Data[1]

Khaled Ben Letaief and John S. Sadowsky
School of Electrical Engineering
Purdue University, West Lafayette, IN 47907

It is typically the case in *large signal* detection that error probability is *exponentially consistent.* Consider the case of a threshold test: T_n is a real valued test statistic derived from n data samples; choose hypothesis H_0 if $T_n < \gamma$ and H_1 if $T_n > \gamma$. Under hypothesis H_i, the *exponential rate* of decrease of error probability is

$$\lim_{n\to\infty} \frac{1}{n} P_i(\text{error}) \;=\; I_i(\gamma) \tag{1}$$

where $I_i(\cdot)$ is called the large deviations *rate function.* The rate function is expressed directly in terms of the *cumulant* function $\mu_i(s) \triangleq \lim_{n\to\infty} (1/n) \ln(E_i[\exp(nsT_n)])$ via a relationship known as the Fenchel transform.

Large deviations theory also provides accurate *finite sample* error probability approximations via a relationship called the representation formula. For the H_0 hypothesis,

$$P_0(\text{error}) \;=\; \exp(-nI_0(\gamma)) \int_\gamma^\infty \exp(-s(t-\gamma))\, dF_{s,n}(t) \tag{2}$$

where the optimal value of s is provided by the Fenchel transform, and $F_{s,n}(t)$ is a distribution function. Moments of $F_{s,n}(t)$ may be obtained by differentiating $\mu_i(s)$. Approximations, upper *and* lower bounds are obtained by approximating the integral in (2). It is well known that this integral behaves like $O(1/\sqrt{n})$, which is obtained by applying weak convergence method to the "tilted" distribution sequence $\{F_{s,n}\}$. A topic of future research will be to investigate maximum entropy approximation, and more importantly, moment bounds for the integral in (3).

In our study, we consider memoryless detectors, $T_n \mp (1/n) \sum_{k=1}^{n} g(Y_k)$, generated by Markov data $\{Y_k\}$. To numerically compute $\mu_i(s)$ we approximate the Markov distributions by a finite state space Markov chain. Three methods are compared: TR) true computation of $\mu_i(s)$ via the the spectral radius of a *transform kernel;* LF) the "least favorable" dependency structure approach as developed by Sadowsky (see reference); and G) Gaussian approximations. The LF approach provides a lower bound for the true rate function, hence error probability estimates obtained using this method should be pessimistic. In fact, our preliminary computations indicate that the LF rate function can very closely approximate the TR rate, and furthermore, the approximation seems to be uniformly good as γ varies. Gaussian approximations, on the other hand, are often hopelessly over optimistic, especially in large signal - moderate sample size situations. TR and LF also give surprisingly accurate finite sample error probability estimates. There are two advantages of the LF method: it is the robust method for uncertain data dependency, and it is an order of magnitude more numerically efficient than the exact method TR.

---

[1] This work is supported by the National Science Foundation, Grant No. DCI-8601071.

For example, a 200 state approximation of a Gaussian AR(1) process with one shift autocorrelation 0.1 plus a constant signal of amplitude of 1.5 was considered. The detector was the simple sign detector, g(y) = sgn(y), with threshold $\gamma = 0$. Figure 1 compares the exact rate function TR with the Gaussian approximation rate function G. The approximate rate function of method LF, which is a lower bound for TR, was indistinguishable from TR on the scale of Figure 1. The table below gives error probability estimates obtained using the simplest maximum entropy approximation for the integral in (2). These are also compared to error probability estimates obtained from Monte Carlo simulations, denoted M.

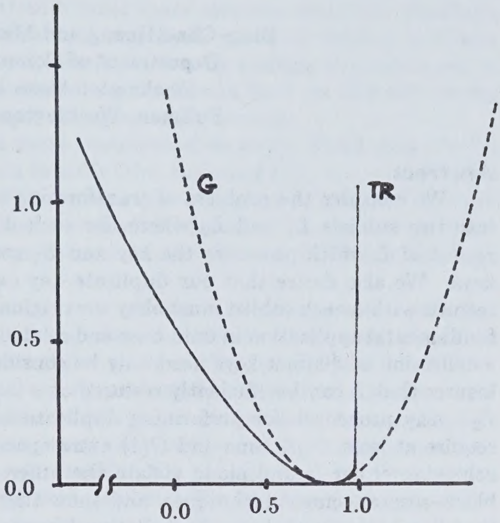


Figure 1.

n =	3	5	7	9
M	.0161	.00399	.00107	.000240
TR	.0225	.00453	.00099	.000226
LF	.113	.00997	.00291	.000883
G	.0007	.000017	.00000046	.000000013

An immediate objection to our comparison of the Gaussian approximation might be that the sample sizes where far too small for central limit theory to provide accurate approximations. However, for a fixed signal amplitude, we have

$$\frac{P(\text{error})}{\hat{P}(\text{error})} \sim \exp(n(I_i(\gamma) - \hat{I}_i(\gamma))) \tag{3}$$

where $\hat{P}$(error) is an error probability approximation with rate function $\hat{I}(\gamma)$. We see that Gaussian approximations are **arbitrarily bad as n gets large!** It is typically the case that the Gaussian rate exceeds the true rate, hence the approximations are exponentially over optimistic. While this fact does run contrary to popular belief, it does not contradict central limit theory because central limit theory does not apply in consistent detection problems.

Reference: John S. Sadowsky, "An asymptotically least favorable Chernoff bound for a large class of dependent data processes," to appear in *IEEE Trans. Inform. Theory*, Jan., 1987.

Stable Duplicate–Key Extraction with Optimal Time and Space Bounds*

Bing–Chao Huang and Michael A. Langston
Department of Computer Science
Washington State University
Pullman, Washington 99164–1210

Abstract.

We consider the problem of transforming a list L of records sorted on some key into two sublists L_1 and L_2 where, for each distinct key in L, L_1 contains the first record of L which possesses the key and L_2 contains all records of L with duplicate keys. We also desire that our duplicate–key extraction algorithm be stable (that is, records within each sublist must obey the original order given by L). This operation has fundamental application in data base and related file processing environments whenever a collection of distinct keys need only be considered. Moreover, stability in extraction insures that L can be efficiently restored at a later time with a stable merge of L_1 and L_2. Any procedure for performing duplicate–key extraction on a file of size n must require at least $O(n)$ time and $O(1)$ extra space, although the obvious algorithms for achieving either bound alone violate the other. We design a stable algorithm, using block–rearrangement techniques, and show that it is optimal in the theoretical sense that it achieves both lower bounds simultaneously. We also prove that its worst–case number of key comparisons and record exchanges sum to no more than $6n$, suggesting that the algorithm has practical application as well.

1. Introduction

Suppose that we are given a list L with n records sorted on some key, and that some records may possess the same key. We may wish to entertain a variety of questions focusing only on distinct keys. For example, one might want to know whether a file contains a particular key, whether two or more files contain the same keys, whether all keys are present within a certain range, etc. Repeated queries of this nature frequently arise in data base and business data processing environments, and can often be most efficiently addressed by first extracting duplicate keys from L, leaving but one copy of each distinct key.

For a file of substantial size, an extra list containing distinct keys as well as their associated records may require a prohibitive amount of additional storage. Therefore, we focus our attention on schemes which extract duplicates *in–place*, so that L is transformed into the concatenation of two sublists L_1 and L_2, where L_1 contains one record for each distinct key in L and L_2 contains $L - L_1$. We allow the use of a few extra memory cells to aid in the transformation, but their total number must be $O(1)$, independent of n. In–place extraction, however, dictates that the operation be invertible, so that we can restore L to its original sequence if necessary. Hence we ask that L_1 contain the *first* copy of each distinct key and that our algorithm be *stable*, by which we mean that keys within each sublist retain the relative order they held within L. Clearly a subsequent in–place, stable merge of L_1 and L_2 restores L (see [Pa] for an efficient implementation of such a merge).

Any algorithm to solve this problem needs at least $O(n)$ time, since every key must be examined at least once to determine whether it is duplicated within L. There is, of course, an obvious way to construct L_1 and L_2 in only $O(n)$ time, by a linear scan of

---

* This research is supported in part by the National Science Foundation under grants MCS–8203868 and ECS–8403859.

L and temporary use of a list of storage cells separate from L. This simple method is, unfortunately, unacceptable since $O(n)$ additional space must be available. Similarly, there is a straightforward way to construct L_1 and L_2 in-place, by finding each new element of L_1 in turn and moving it to its final position by shifting duplicates out of the way. In this case, however, the interested reader should have no difficulty seeing that as much as $O(n^2)$ time may be required just to move records.

In the sequel we will show that a more complicated strategy, which uses $O(\sqrt{n})$ blocks of size $O(\sqrt{n})$, solves the problem in both $O(n)$ time and $O(1)$ space, and hence is optimal to within a constant factor. This general type of approach, in which blocks of size $O(\sqrt{n})$ are exchanged, has been shown to be useful for other file operations (most notably sorting and merging [Pa]) and can be traced back to the seminal work described in [Kr].

In the following section, we introduce some necessary notation and define a few useful, primitive suboperations. Section 3 contains an overview of the main algorithm and an example, with simplifying assumptions made on list, sublist and block sizes in order to facilitate discussion. Time and space bounds are derived. In Section 4 we describe minor implementation details which permit the simplifying assumptions to be dropped. We also show that the worst-case number of key comparisons and record exchanges sum to no more than $6n$, making the algorithm of practical as well as theoretical interest. In Section 5, we turn our attention to average-case behavior and report on computational experiments which suggest that, for uniformly distributed duplicates, the expected penalty for reducing additional storage from $O(n)$ to $O(1)$ is less than a fourfold increase in program execution times. The final section contains a discussion of a few conclusions which can be drawn from this effort.

2. Notation and Preliminaries

Let $L = < R_1, R_2, \ldots, R_n >$ denote a list of n records, where a record of index i, $1 \le i \le n$, has m data fields, taking the form $R_i = < F_{i1}, F_{i2}, \ldots, F_{im} >$. We assume that for some unique k, $1 \le k \le m$, all records are sorted in non-decreasing order on field k so that $F_{1k} \le F_{2k} \le \ldots \le F_{nk}$. We henceforth refer to this kth field as the *key*, and use $KEY(i)$ as a shorthand to denote field k of record i. For the sake of complete generality, we allow neither the key nor any other field of a record to be modified during duplicate extraction. We use the term L_1 record to denote one bound for L_1. That is, an L_1 record is either R_1 or some $R_i, 1 < i \le n$, where $KEY(i-1) < KEY(i)$. We employ the term L_2 record in an analogous fashion to denote one bound for L_2.

Only the two common $O(1)$ time and space file operations are assumed, namely, record exchanges and key comparisons. The exchange procedure, $SWAP(i, j)$, directs that the ith and jth records are to be exchanged. The comparison functions, for example $KEY(i) < KEY(j)$, return the expected Boolean values dependent on the relative values of the keys being compared.

From these primitive operations we construct a few $O(1)$ space useful subprograms for dealing with *blocks*. Let us define a block to be a set of records from L with consecutive indices. The *head* of a block is the record with lowest index; the *tail* is the one with the highest index. If a block contains only L_1 (L_2) records, then we may refer to it as an L_1 (L_2) block. The procedure $BLOCKSWAP(i, j, h)$ exchanges a block of h records beginning at index i with a block of h records beginning at index j in $O(h)$ time. We specify that blocks do not partially overlap (i.e., if $i \ne j$ then $h \le |i-j|$) and that when $BLOCKSWAP$ is finished, records within a moved block retain the order they possessed before $BLOCKSWAP$ was invoked. A block of h records beginning at index i is sorted in non-decreasing order by the procedure $SORT(i, h)$. Finally, the procedure $BLOCKSORT(i, h, p)$ uses $BLOCKSWAP$ to rearrange the p consecutive blocks, each with h records, beginning at index i so that their heads are sorted in non-decreasing order. To reduce unnecessary record movement, which is important should m be relatively large, we assume *SORT* and *BLOCKSORT* use a straight selection sort [Kn], yielding respective time complexities $O(h^2)$ and of $O(p^2 + ph)$.

3. An Overview of the Main Algorithm

In order to facilitate discussion, let us assume for the moment that n is a perfect square, with $a\sqrt{n}$ L_1 records and $b\sqrt{n}$ L_2 records, where a and b are positive integers and $a + b = \sqrt{n}$.

The first step of the algorithm is to fill an "internal buffer" of size $\sqrt{n}$ with the $\sqrt{n}$ largest-keyed L_1 records. Thus we convert L into the form ABC where the records of A have not been disturbed, B is the buffer, and C is a suffix (rightmost sublist) of L_2. B is constructed by conducting a right-to-left scan of L. When a comparison of adjacent keys reveals that a new L_1 record has been found, it is coalesced into B. When an L_2 record is encountered, it is exchanged with the current rightmost element of B. Therefore, B begins with size zero at the right end of L and grows as we "roll" it toward the left until it accumulates $\sqrt{n}$ L_1 records, now unordered by key. For simplicity, we assume that the size of A, and hence C, is an integral multiple of $\sqrt{n}$.

It should be clear that, at this point, no more than n key comparisons and n record exchanges are needed. Also, a couple of additional storage cells are enough to keep track of the buffer's position.

The second step of the algorithm is the most complex. AB is transformed into XY, where X contains a L_1 blocks (one of which is B) and $YC = L_2$. This is accomplished as follows. B is used to partition A into a collection of $\sqrt{n}$-sized L_1 and L_2 blocks. B's initial position for this step will become the rightmost L_2 block of A. The $\sqrt{n}$ memory cells to its immediate left will become an L_1 block. A is scanned from right-to-left until an L_2 record is found, which is then exchanged with the buffer's tail. The scan is continued, each record in turn being exchanged with the rightmost buffer element in the appropriate block. Thus the buffer is, in general, broken into two pieces, each to the left of the growing edge of a block. When an L_1 block is filled, a new L_1 block is begun in the set of $\sqrt{n}$ cells to its immediate left. When an L_2 block is filled, *BLOCKSWAP* is invoked if necessary to move it to its final position, ousting an L_1 block there. Observe that handling L_2 blocks in this manner insures that Y will contain the appropriate prefix (leftmost sublist) of L_2. A new L_2 block is begun in the position of the leftmost L_1 block, after its c L_1 records, $0 \leq c < \sqrt{n}$, are exchanged with the rightmost c records, all from B, in the block to its immediate left. (Therefore a new L_2 block initially contains all of B.) When all of A has been scanned in this manner, every block of X, except for B, remains sorted internally. The blocks themselves may, however, be unordered with respect to each other.

At most n comparisons are needed to compare adjacent keys. $O(n)$ time suffices for record movement in this step, since there are at most n record exchanges and at most $2\sqrt{n}$ block exchanges. Only a small, constant number of additional storage cells are needed for counters and pointers.

The final step of the algorithm is to transform X into L_1. To do this, B is first sorted (recall that all keys in B are distinct, insuring stability). Then *BLOCKSORT* is used to sort the blocks of X by their heads. Since these blocks were constructed one at a time, and since each is sorted internally, L_1L_2 is the final result.

$O(n)$ time and $O(1)$ space are sufficient for this step as well, since each sort involves at most $\sqrt{n}$ keys.

4. Implementation Details

We now give a complete version of the algorithm, describing implementation details which permit us to perform duplicate–key extraction without regard to any assumptions about actual list, sublist or block sizes. Let $s = \lfloor\sqrt{n}\rfloor$ denote the size we will use for a block. Thus the number of L_1 records is $st_1 + e_1$, for some unique non–negative integers t_1 and e_1, where $e_1 < s$. Similarly, the number of L_2 records can be denoted by the expression $st_2 + e_2, e_2 < s$. Note that $(s+1)^2 > n \geq st_1 + st_2$ guarantees that the total number of s–sized blocks, $t_1 + t_2$, is no more than $s + 2$. We now consider the file as a collection of $t_1 + t_2 + 2$ blocks, the first a (possibly empty) block of size e_1, followed by the $t_1 + t_2$ s–sized blocks, followed by a final (possibly empty) block of size e_2.

First, we attempt to fill the internal buffer. Observe that in doing so we may have examined all L_2 records, in which case we merely sort the (perhaps only partially filled) buffer and halt. Otherwise, we scan the remainder of L to determine the number of L_1 records, from which we derive the value of e_1. The block containing the rightmost buffer element becomes the first L_2 block, the block to its immediate left the first L_1 block. We are thus either finished with the last block, which is of size e_2, or soon will be. Its records will end up in their final position; its unusual size can cause no problem.

We initiate the second step of the algorithm as outlined in the previous section, using *block* to denote the beginning address of the current L_2 block and *ptr*1 (*ptr*2) as a pointer to the location which is to receive the next L_1 (L_2) record. When the last L_2 block has been filled and moved, we know there is no need to continue scanning A. Note that, with the final exchange of fewer than s L_1 and buffer records, the buffer now occupies one full block, ensuring that we are finished with the first block (which has the unusual size e_1) as well as any block to the left of the buffer.

We complete the job of duplicate–key extraction by sorting and moving the buffer to its final position, and then invoking *BLOCKSORT* on the L_1 blocks which may need rearrangement. Hence the following pidgin Algol description of algorithm *EXTRACT* and its two internal procedures *BUFFERFILL* and *BLOCKIFY*.

```
Algorithm EXTRACT (L, n)
begin
    $s \leftarrow \lfloor\sqrt{n}\rfloor$
    BUFFERFILL
    if buffer = 2
        then SORT (2, size) and return (L)
    new ← buffer −1
    count ← $s$ +1
    for $i$ ← 2 to new do
        if KEY ($i$ − 1) < KEY ($i$)
            then count ← count +1
        end for
    $e_1$ ← count mod $s$
    BLOCKIFY
    SORT (buffer, $s$)
    BLOCKSWAP (buffer, count −$s$ + 1, $s$)
    BLOCKSORT (buffer, $s$, (count +1 − buffer) div $s$ − 1)
    end EXTRACT.
```

```
procedure BUFFERFILL
```

```
begin
    buffer ← $n+1$
    size ← 0
    repeat
        buffer ← buffer −1
        if KEY (buffer −1) = KEY (buffer)
            then SWAP (buffer, buffer + size)
            else size ← size + 1
        until buffer = 2 or size = s

    end BUFFERFILL.


procedure BLOCKIFY
begin
    block ← new + s − (new − $e_1$ − 1) mod $s$
    ptr1 ← block −1
    ptr2 ← new + $s$
    for $i$ ← block downto count +1 step −$s$ do
        repeat
            while KEY (new −1) < KEY (new) do
                SWAP (new, ptr1)
                new ← new −1
                ptr1 ← ptr1−1
                end while
            SWAP (new, ptr2)
            new ← new −1
            ptr2 ← ptr2 −1
            until ptr2 = $s \cdot ((\text{ptr2}-e_1)$ div $s)+e_1$
        BLOCKSWAP (ptr2 + 1, $i$, $s$)
        ptr1 ← new
        ptr2 ← new + 2$s$ − (new - $e_1$ − 1)mod $s$ − 1
        BLOCKSWAP (new +1, new + $s+1$, ptr2–new–s)
        end for
    buffer ← ptr2 − $s$ + 1
    end BLOCKIFY.
```

We now bound the worst–case number of key comparisons and record exchanges performed by *EXTRACT*. Exactly $n-1$ comparisons are employed to extract the buffer and to count the number of L_1 records. *BLOCKIFY* requires no more than $n-s-1$ comparisons, all within its while loop. Our selection sort implementation of *SORT* can direct at most $.5s(s-1)$ comparisons [Kn]. Similarly, *BLOCKSORT* needs no more than $.5(t_1-1)(t_1-2) \leq .5(s+1)(s)$ comparisons. Therefore, the total number of key comparisons is at most $(n-1)+(n-s-1)+.5s(s-1)+.5(s+1)s = 2n+s^2-s-2 \leq 3n-s-2 < 3n$.

As for record exchanges, *SWAP* is invoked at most $n-s-1$ times. Each *BLOCKSWAP* within *BLOCKIFY* is called at most t_2 times, always with block size less than or equal to s. The main algorithm's final *SORT* requires $s-1$ exchanges, followed by *BLOCKSWAP* needing s. Finally, *BLOCKSORT* uses $s(t_1-2)$ exchanges. Therefore, the total number of record exchanges is at most $(n-s-1)+2st_2+$

$(s-1)+s+s(t_1-2) = n+s(t_1+t_2)+st_2-s-2 \leq 3n-s-2 < 3n.$

Since key comparisons and record exchanges are likely to be by far the most time consuming operations for $EXTRACT$, and since they are both storage–to–storage type instructions for most architectures, we conclude that $6n$ is a reasonable estimate of the worst–case constant of proportionality for this algorithm's $O(n)$ time complexity. As for $O(1)$ space, a review of our code reveals that we have used but 10 additional storage cells for pointers and counters.

5. Computational Experience

As a more complete investigation of $EXTRACT$'s practicality, we briefly describe the results of a series of experiments conducted to compare it to a naive but efficient algorithm, $NAIVE$, which is free to exploit the temporary use of $O(n)$ extra memory. This simple algorithm conducts a left–to–right scan of L, moving each L_1 record to its final position and moving each L_2 record to temporary storage. After the scan is completed, the L_2 records are moved to their final positions.

We wrote both algorithms in Pascal for a VAX 750, and timed them over large collections of keys, with duplicates uniformly chosen at random. Table 1 summarizes the behavior we observed. Average runtimes reported (A_E and A_N) reflect, for each list size n, the mean execution time for 44 experiments with varying duplicate densities: four lists with no duplicates, four with 10% duplicates, four with 20% duplicates and so on up to four with 90% duplicates, and even four with but a single key value, which approaches having 100% duplicates. While the ratios reported in Table 1 are useful, we note that the performance of $EXTRACT$ relative to $NAIVE$ is dependent on the exact percentage of duplicates encountered. Therefore we use $n = 1,000$ as a representative list size and empirically illustrate this relationship in Figure 1.

Naturally, other factors such as record length, computer architecture, and so on affect algorithmic behavior. Therefore, the ratios we derive can only be viewed as (conservative) estimates. We conclude that the expected penalty for performing duplicate–key extraction in place is less than a quadrupling of program execution times (generally, a lot less). We emphasize that, for the sake of simplicity and straightforwardness, we have refrained from any attempt to "tinker around" with $EXTRACT$ to make it faster. We conjecture that a more streamlined implementation can perform duplicate–key extraction at a considerably accelerated pace.

6. Conclusions

We have devised an algorithm which performs stable, duplicate–key extraction in linear time and constant space, and is therefore optimal to within a constant factor. We have also analyzed its worst–case number of key comparisons and record exchanges, as well as conducted a series of computational experiments, which indicate that it possesses a marked potential for practicality.

The interested reader may have already observed that this algorithm could be viewed as an efficient means for increasing the effective size of internal memory when performing duplicate–key extraction on a much larger external file. This tends to decrease the number of input/output operations needed, thereby dramatically reducing the overall execution time. Similarly, this algorithm could be employed to great advantage when managing critical resources such as cache memory or other relatively small, high–speed memory components.

We remark that block exchange techniques may well be applied advantageously to an entire host of file permutation problems. Although their general effectiveness remains to be determined, we suggest that they hold promise for more widespread use as theoretical and practical algorithmic design tools.

References

[Kn] D. E. Knuth, *The Art of Computer Programming, Vol 3: Sorting and Searching,* Addison–Wesley, Reading, MA, 1973.

[Kr] M. A. Kronrod, "An Optimal Ordering Algorithm Without a Field of Operation," Dok. Akad. Nauk SSSR 186 (1969), 1256–1258.

[Pa] L. T. Pardo, "Stable Sorting and Merging with Optimal Space and Time Bounds," SIAM J. Comput. 6 (1977), 351–372.

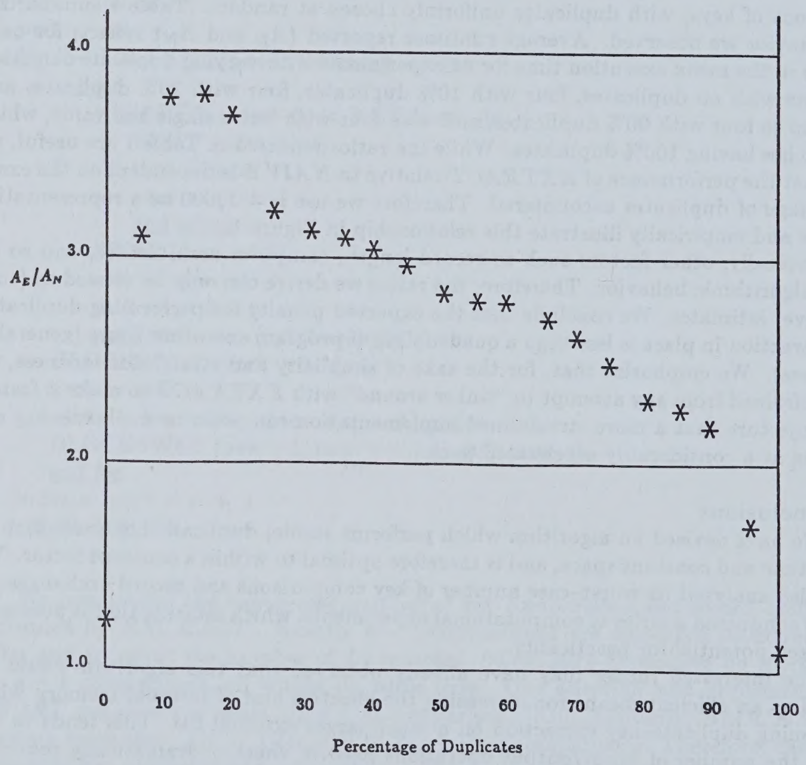


Figure 1. EXTRACT's performance relative to NAIVE, as a function of duplicate density. $n = 1{,}000$.

List Size, n	EXTRACT's Average Runtime, A_E, in milliseconds	NAIVE's Average Runtime, A_N, in milliseconds	A_E/A_N
50	9.1	4.1	2.22
100	23.3	9.55	2.43
500	109	46.1	2.36
1,000	235	92	2.55
5,000	1186	450	2.64
10,000	2440	912	2.68
50,000	12,579	4,670	2.69
100,000	25,529	9,575	2.67

Table 1. Observed behavior of EXTRACT relative to NAIVE.

ARRAY ACCESS BOUNDS FOR BLOCK STORAGE MEMORY SYSTEMS

ARVIN PARK
K. BALASUBRAMANIAN
RICHARD J. LIPTON

Department of Computer Science
Princeton University
Princeton, New Jersey 08544

ABSTRACT

Paging performance can be a dominant factor in a program's running time. Many seemingly efficient data structures and algorithms lose orders of magnitude in performance because they generate an excessive number of page faults. This paper shows that tradeoffs exist between average row access speed S_r (which is defined as number of row elements retrieved divided by the number of blocks accessed) and average column access speed S_c (defined similarly). We prove the S_rS_c product is optimally bounded by the block size N, and generalize to other access patterns. Practical array access strategies are developed and extensions to these results are discussed.

Keywords - *arrays, paging performance, secondary storage device.*

INTRODUCTION

Paging performance can be a dominating factor in an application's running time [1]. Many seemingly efficient data structures and algorithms lose orders of magnitude in performance because they generate an excessive number of page faults. Several papers have been written on array storage schemes in paged memory systems [2][3][4]. McKellar and Coffman [2] noted the performance advantages of storing submatricies on pages instead of the standard row or column major formats (Figure 1). Fischer and Probert [3] have developed efficient methods of reorganizing a row or column oriented matrix into submatrix form. This paper extends these results by showing that tradeoffs exist between average row access speed S_r (which is defined as number of row elements retrieved divided by the number of blocks accessed) and average column access speed S_c (defined similarly). We prove the S_rS_c product is optimally upper

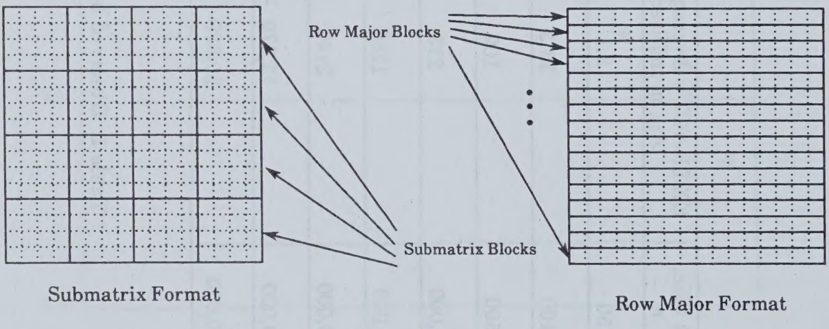


Figure 1

bounded by the block size N. Practical array access strategies are developed and extensions to these results are discussed.

MODEL

Block storage devices span an entire range of block sizes and access times: Cache lines and interleaved semiconductor stores support block sizes from one to several dozens of words and have access times measured in hundreds of nanoseconds; Rotating storage supports block sizes of hundreds to thousands of words and access times in the tens of milliseconds; RAM disks, CCD, and magnetic bubble stores fit between these two extremes in block sizes and access times.

To analyze array storage schemes for all of these different systems we have developed a simple model. We assume a block memory system is a set of L storage "blocks" of exactly N data elements (words). We will measure performance by the number of "page pulls" required by an application to perform its task.

We define **data set** as the entire collection of data elements (words) which is stored in the memory system. We define an **allocation** as an assignment of elements of the data set onto blocks of the storage device. We define an **access vector** as a subset of the data set which we desire to access. For example columns or rows of an array would be common access vectors.

Arrays are assumed to be two dimensional although extensions to further dimensions can be made. Performance on several types of access vectors will be evaluated; Rows, Columns, Diagonals, Back-Diagonals, and Rectangular Blocks. Although other many more access vectors are possible these are by far the most prevalent [6][8][9].

ALLOCATIONS

To achieve maximal performance it is necessary to allocate the array elements to blocks of the storage device minimizing the total number of blocks referenced on a set of access vectors. This is difficult even if the set of access vectors are known in advance. For a fixed allocation, the number of vectors of a given size which fit exactly onto the minimum number of blocks is a great deal smaller than the number of possible vectors.

Assume an access vector is of size V, and that N (the number of elements in a block) divides V. The entire data set consists of KN elements and fits neatly into K blocks of the storage device. For a fixed allocation there exists

$$\binom{K}{V/N}$$

possible access vectors of size V that fit exactly into K blocks. This is in contrast to

$$\binom{KN}{V}$$

possible vectors of size V. The ratio of these two quantities is:

$$\frac{\binom{K}{V/N}}{\binom{KN}{V}} = \frac{\dfrac{K!}{(K-V/N)!(V/N)!}}{\dfrac{KN!}{(KN-V)!\,V!}} = \frac{K!\,V!\,(KN-V)!}{KN!\,(V/N)!\,(K-V/N)!}$$

Using Stirling's Approximation this becomes:

$$\approx \frac{(2\pi K)^{1/2}\left(\frac{K}{e}\right)^{K}(2\pi V)^{1/2}\left(\frac{V}{e}\right)^{V}(2\pi(KN-V))^{1/2}\left(\frac{KN-V}{e}\right)^{(KN-V)}}{(2\pi KN)^{1/2}\left(\frac{KN}{e}\right)^{KN}(2\pi V/N)^{1/2}\left(\frac{V/N}{e}\right)^{V/N}(2\pi(K-V/N))^{1/2}\left(\frac{K-V/N}{e}\right)^{(K-V/N)}}$$

$$= N^{(1/2 + V/N - V)}\, K^{(-KN+1)}\, V^{(V-V/N)}\, (K-V/N)^{(KN - V - K + V/N)} \approx \left(\frac{V}{KN}\right)^{V}$$

$$= \left(\frac{SIZE\ OF\ VECTOR}{SIZE\ OF\ DATA\ SET}\right)^{(SIZE\ OF\ VECTOR)}$$

This is a very small number if the data set is much larger than the desired vector.

ROW-COLUMN BOUND

There is a tradeoff between facility in accessing rows and in accessing columns. For example, if an array is stored in row major form (blocks contain adjacent row elements) rows can be accessed very quickly. For each block accessed all the elements can be used. At the same time column access becomes slow (For each block accessed only one element can be used). This tradeoff can be bounded. We define **average row access speed** as an average over all rows in the array, of the number of elements in a row divided by the number of blocks which contain elements of the row. If a row was of size N and was contained entirely on one block of size N, the access speed for that row would be N. If the same row was contained on N separate blocks the row access speed would be 1. **Average column access speed** is defined analogously.

For any two dimensional array A that is partitioned into blocks of size N without replication of elements, i.e., each element is stored in one and only one block (it should be obvious that storing the same element twice in the same block cannot be of any advantage), the average row access speed S_r multiplied by the average column access speed S_c is less than or equal to N. $S_rS_c \geq N$. This not only applies to rows and columns, but any two **orthogonal partitions** of the array: A **partition** K of a data set D, is a set of disjoint subsets of D that spans D (an individual disjoint subset is called a **member** of K). Two partitions A,B of a data set D are said to be **orthogonal** iff $\forall a_i \in A \wedge \forall b_j \in B, a_i \cap b_j = \{d\}, d \in D$ or $a_i \cap b_j = \varnothing$, and $a_i \cap b_j = a_k \cap b_l \Leftrightarrow i=k$ and $j = l$. Two partitions, A and B, will be called **completely orthogonal** iff they are orthogonal and $\forall a_i \in A \wedge \forall b_j \in B, |a_i \cap b_j| = 1$. We now establish Theorem 1.

Theorem 1: Given any two orthogonal partitions A, B of a data set D and an allocation of D onto blocks of size N without replication. The average partition access speed on A 'S_A' multiplied by the average partition access speed on B 'S_B' is bounded by N. $S_AS_B \leq N$.

We first establish two lemmas:

Lemma 1: $\forall\, x,y \in Z^+,\ x/y + y/x \geq 2$

Proof:

$$(x-y)^2 \geq 0 \Rightarrow x^2 - 2xy + y^2 \geq 0 \Rightarrow x^2 + y^2 \geq 2xy \Rightarrow \frac{x^2+y^2}{xy} \geq 2 \Rightarrow \frac{x}{y} + \frac{y}{x} \geq 2$$

Lemma 2: For any set of p positive integers x_i, $(i=1,2,3 \ldots p)$

$$\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{x_j}{x_i} \ge p^2$$

Proof:

$$\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{x_j}{x_i} = \frac{1}{2}\left(2\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{x_j}{x_i}\right) = \frac{1}{2}\left(\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{x_j}{x_i} + \frac{x_i}{x_j}\right)$$

By Lemma one:

$$\ge \frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{p} 2 = \frac{1}{2}(2)\,p^2 = p^2$$

We can now proceed with the proof of the theorem. Let M be the number of blocks in data set D, where each block is of size N. Let A and B (e.g., rows and columns) be two orthogonal partitions of D. Each block i of data set D intersects r_i different members of A (rows) and c_i different members of B (columns). An intersection between a block i and a partition element of A is called an **A-fragment of i**. Let $k_i = N/r_i$ and $j_i = N/c_i$.

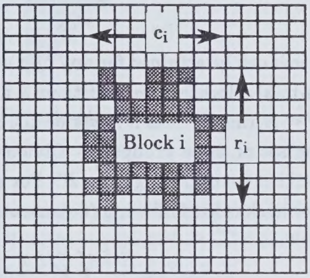


Figure 2

The orthogonality property of A and B, implies the following:

$$r_i c_i \ge N$$

Suppose this were not true:

$$r_i c_i < N$$

The sum of all of the r_i A-fragments of a block i equals N, this implies that the average fragment is of size greater than c_i. By orthogonality each A-fragment and B-fragment can intersect in at most one element. So block i must intersect more than c_i members of B. Contradiction! We now establish two inequalities.

$$r_i c_i \ge N \Rightarrow N/r_i \le c_i \Rightarrow k_i \le c_i \tag{1}$$

$$r_i c_i \ge N \Rightarrow N/c_i \le r_i \Rightarrow j_i \le r_i \tag{2}$$

We now proceed with the main proof.

$$S_A = \frac{\textit{Data Set Size}}{\textit{Number of Fragments}} = \frac{NM}{\sum_{i=1}^{M} r_i} \qquad S_B = \frac{\textit{Data Set Size}}{\textit{Number of Fragments}} = \frac{NM}{\sum_{j=1}^{M} c_j}$$

$$S_A\, S_B = \frac{N^2M^2}{\sum_{i=1}^{M} r_i \sum_{j=1}^{M} c_j} = \frac{N^2M^2}{\sum_{i=1}^{M} \frac{N}{k_i} \sum_{j=1}^{M} c_{j_i}}$$

Using inequality number 1:

$$\leq \frac{N^2M^2}{\sum_{i=1}^{M} \frac{N}{k_i} \sum_{j=1}^{M} k_j} = \frac{N^2M^2}{N \sum_{i=1}^{M} \frac{1}{k_i} \sum_{j=1}^{M} k_j} = \frac{N^2M^2}{N \sum_{i=1}^{M} \sum_{j=1}^{M} \frac{k_j}{k_{i_i}}}$$

Using Lemma 2:

$$\leq \frac{N^2M^2}{NM^2} = N$$

Done.

This bound is optimal since it can be attained for any pair of completely orthogonal partitions of a data set. Produce an optimal allocation for partitions *A* and *B* of data set *D* in the following manner. Find a number $C < N$ that divides *N*. Divide the members (rows) of *A* into disjoint groups of size *N/C*. Call the set of these groups A^*. Divide the members (columns) of *B* into disjoint groups of size *C*. Call the set of these groups B^*. For each pair (x,y), where $x \in A^*$ and $y \in B^*$, Produce a block $G_{x,y}$ which has the following *N* elements.

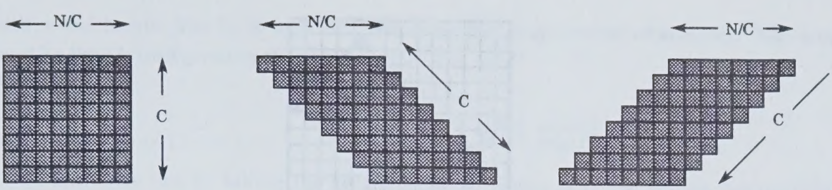
$$G_{x,y} = \bigcup_{}^{\forall i \in x \wedge \forall j \in y} i \cap j$$

Now each member of *A* that we want to access is distributed in fragments of size *N/C* across different blocks of the allocation. Members of *B* are similarly distributed in fragments of size *C*. The *A* access speed "S_A" is therefore $S_A = N/C$, and the *B* access speed, $S_B = C$. The product of these two, $S_A S_B = N$, which meets the upper bound of Theorem 1. This upper bound is therefore optimal.

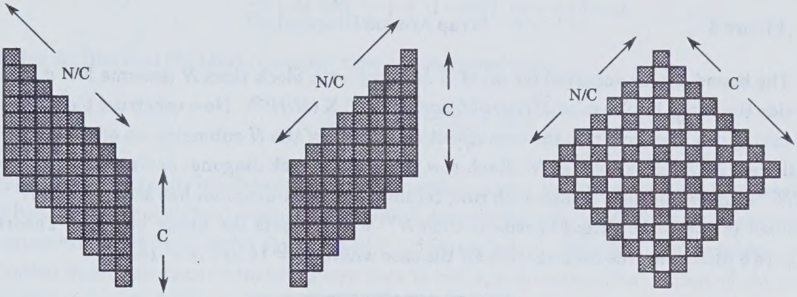
Rectangular blocks achieve this upper bound for the row-column tradeoff. Various diamonds and parallelograms achieve the bound for combinations of rows, diagonals, columns, and back-diagonals (Figure 3). Diagonals and back diagonals intersect differently than the other combinations of rows, columns, diagonals, and back-diagonals. This is because a given diagonal does not intersect every back diagonal it crosses. A matrix is divided into two disjoint sets of diagonals and back diagonals that intersect like the red and black squares of a checker board.

THREE WAY ACCESS BOUNDS

How does this bound extend to accesses along three different partitions of the array? (for instance rows, columns, and diagonals) A bound can be arrived at by noting that each pair of access vectors must be constrained by the pairwise bound $S_A S_B \leq N$. This gives us Theorem 2,



Row-Column
Row-Diagonal
Row-Back Diagonal



Column-Diagonal
Column-Back Diagonal
Diagonal-Back Diagonal

Figure 3

Theorem 2: Given any any three orthogonal partitions *A*, *B*, and *C*, of a data set *D*, and an allocation of *D* onto blocks of size *N* without replication. The product of the average partition access speeds on *A*, *B*, and *C* in bounded by $N^{3/2}$. $S_A S_B S_C \leq N^{3/2}$.

Proof: Assume we have three orthogonal partitions *A*, *B*, and *C* of data set *D*. From Theorem one we have $S_A S_B \leq N$, $S_A S_C \leq N$, and $S_B S_C \leq N$. These imply:

$$S_A \leq N/S_C,\ S_B \leq N/S_C,\ S_A \leq N/S_B, \text{ and } S_C \leq N/S_B \rightarrow$$

$$S_A S_B S_C \leq (N/S_C)(N/S_C)S_C = N^2/S_C,\ S_A S_B S_C \leq (N/S_B)S_B(N/S_B) = N^2/S_B \rightarrow$$

Multiplying we get:

$$(S_A S_B S_C)^2 \leq (N^2/S_C)(N^2/S_B) = N^4/S_B S_C \leq N^3 \rightarrow$$

$$S_A S_B S_C \leq N^{3/2}$$

This bound can be achieved for rows, columns and wrap-around diagonals. Wrap around diagonals are diagonals that intersect every row and column of the array. More precisely, given an array element (*R*,*C*), it belongs to a wrap-around diagonal consisting of array elements ((*R* + *I*) mod row size, (*C* + *I*) mod column size) for all integers *I* (Figure 4).

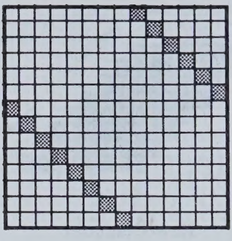


Figure 4 Wrap Around Diagonal

The bound can be achieved for an *M* X *M* array with block sizes *N* (assume $N^{1/2}$ divides *M*). Divide the array into *N* submatricies of size $(M/N^{1/2})$ X $(M/N^{1/2})$. Now construct blocks of size *N* by taking one element from the same position in each of the *N* submatricies of the array. This will yield M^2/N blocks of size *N*. Each row, column, or back diagonal of size *M* is contained on $M/N^{1/2}$ different blocks. Hence each row, column, and back-diagonal has access speed $N^{1/2}$. The product of the three access speeds is then $N^{3/2}$ which meets the upper bound of Theorem 2. Figure 5 illustrates the construction for the case where $M = 16$ and $N = 16$.

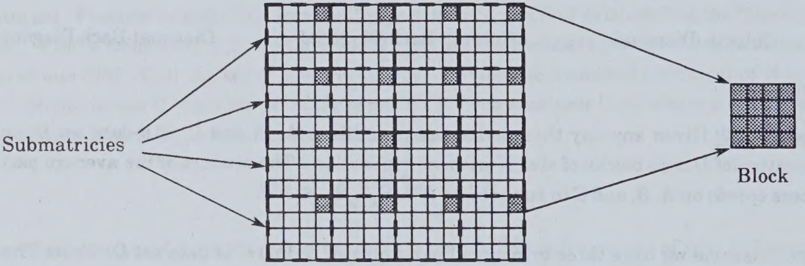


Figure 5 Optimal Three Way Bound Construction

We have not been able to achieve the $N^{3/2}$ bound for conventional diagonals, rows and columns. $N^{1/2}$ X $N^{1/2}$ Submatrix blocks achieve a $N^{3/2}/2$ product for diagonal, row, and column access speeds which is within a factor of two of the upper bound in Theorem 2. We conjecture that the $N^{3/2}$ product cannot be achieved for row, column, and diagonal access speeds. This bound will be the focus of future research.

ARRAY ACCESS STRATEGIES

The row-column speed bound suggests strategies for array storage if the relative frequencies of row and column accesses are known beforehand. For example, if a particular application accesses random rows and columns. The row accesses occur with probability α and column accesses occur with probability 1 - α. We can produce an allocation that minimizes the total execution time.

Let the row size be $R >> N$ and column size be $C >> N$. *N* is the block size. We will allocate rectangular sections of the array onto different blocks. Assume these blocks are of width w and length N/w ($1 \le w \le N$). Let A be the total number of accesses. The total execution time is then given by the expression.

$$A\left(\alpha R\left(\frac{1}{w}\right) + (1-\alpha)C\left(\frac{1}{N/w}\right)\right)$$

We minimize this by taking the derivative with respect to w and setting the resulting expression equal to zero.

$$\frac{\delta}{\delta w}\left(A\left(\alpha R\left(\frac{1}{w}\right) + (1-\alpha)C\left(\frac{1}{N/w}\right)\right)\right) = 0$$

From this we find that the total execution time is minimized when:

$$w = \left(\left(\frac{\alpha}{1-\alpha}\right)\left(\frac{R}{C}\right)N\right)^{1/2}$$

Let us take an example involving data motion between cache and main memory. suppose the cache has a line size of sixteen words. Assume that data can be referenced from the cache every 100 nanoseconds while a cache fault takes 1 microsecond to process. Also assume that instruction execution time is insignificant (this is not a bad assumption if most of the code resides in a separate instruction cache). Suppose rows and columns are of equal length, and that rows are referenced half of the time and columns are referenced half of the time. Using the formula for w above, we see that the blocks should be of width four and length four. Let's compare this storage scheme to the conventional row major scheme.

Average access time per word (row major)
= 1/2(avg. word access time for row) + 1/2(avg. word access time for column)
= 1/2((1/16)1000ns + (15/16)100ns) + 1/2(1000ns) = **578.125ns**

Average access time per word (4X4 blocks)
= 1/2(avg. word access time for row) + 1/2(avg. word access time for column)
= 1/2((1/4)1000ns + (3/4)100ns) + 1/2((1/4)1000ns + (3/4)100ns) = **325ns**

The new storage scheme gained a factor of two in performance! Of course the row column access patterns become more complicated, and this leads to more complicated coding or compilation, but in some applications this is the only way to achieve optimal performance.

The benefits of this type of storage become even more pronounced as the block sizes increase. In computations that require numerous accesses to RAM disk, CCD memory, or rotating storage devices even larger gains can be realized.

A NOTE ON DIAGONALS

It should be noted that diagonals/back-diagonals are not completely orthogonal to rows/columns even though (see figure 3) it seems that we have a parallelogram shape that seems to tile the plane and achieve the bound of N for the product of row-diagonal speedups. This is because the arrays we are dealing with are rectangular so diagonals and back-diagonals do not fit "neatly" along the boundaries of the array. These 'edge effects' lead to a speedup somewhat smaller than that which can be achieved for rows and columns. These edge effects diminish in importance as the arrays get larger , and in fact for arrays that extend to infinity on all four sides, rows and diagonals are *completely* orthogonal ! However even with rows (& columns) being completely orthogonal to diagonals (back-diagonals) for 'infinite' arrays, it still does not seem possible to achieve more than $N^{3/2}/2$ for the product of row-column-diagonal speedup. It is also noteworthy that diagonals and back-diagonals do not form a pair of completely orthogonal partitions of the array elements but rather *two pairs of orthogonal partitions of two disjoint subsets of the array elements!* This seems to indicate some basic structural differences between rows/columns and diagonals/back-diagonals which needs to be investigated further.

CONCLUSIONS AND FUTURE RESEARCH

We have shown that the product of access speeds for any two of (row, column, diagonal, back-diagonal) is optimally upper bounded by the block size N. This was extended to show that the product any three access speeds (row, column, diagonal, back-diagonal) is upper bounded by $N^{3/2}$. This three way bound can be met for rows, columns, and wrap around diagonals, but it remains and open question if this bound can be met for rows columns and regular diagonals.

Future research will involve extending these bounds to multidimensional arrays and examining the advantages of redundant storage (storing a single array element on more than one block).

REFERENCES

[1] A. Park. H. Garcia-Molina, "Performance Through Massive Memory", Internal Report, Department of Computer Science, Princeton University, June 1986.

[2] A. C. McKellar, E. G. Coffman, Jr., "Organizing Matricies and Matrix Operations for Paged Memory Systems", *Communications of the ACM*, Volume 12, Number 3, March 1969.

[3] P. C. Fischer, R. L. Probert, "Storage Reorganization Techniques for Matrix Computation in a Paging Environment", *Communications of the ACM*, Volume 22, Number 7, July 1979.

[4] C. B. Moler, "Matrix Computations with Fortran and Paging", *Communications of the ACM*, Volume 15, Number 4, April 1972.

[5] K. E. Batcher, "The Multidimensional Access Memory in STARAN," *IEEE Transactions on Computers*, Volume C-26, Number 2, pp. 174-177, February 1977.

[6] P. Budnick, D. J. Kuck, "The Organization and Use of Parallel Memories", *IEEE Transactions on Computers*, Volume C-20, Number 12, pp. 1566-1569, December 1971.

[7] T. Feng, "Data Manipulating Functions in Parallel Processors and Their Implementations", *IEEE Transactions on Computers*, Volume C-23, Number 3, pp. 309-318, March 1974.

[8] D. H. Lawrie, "Access and Alignment of Data in an Array Processor," *IEEE Transactions on Computers*, Volume C-24, Number 12, pp. 1145-1155, December 1975.

[9] D. H. Lawrie, "The Prime Memory System for Array Access," *IEEE Transactions on Computers*, Volume C-31, Number 5, pp. 134-141, May 1982.

[10] C. V. Ravi, "On the Bandwidth and Interference in Interleaved Memory Systems," *IEEE Transactions on Computers*, Volume C-21, Number 8, August 1972.

A NOTE ON LOWER BOUNDS FOR MIN-MAX HEAPS[*]

A. Hasham and J.-R. Sack
School of Computer Science
Carleton University
Ottawa, Ontario K1S 5B6

Summary. The *heap* was originally introduced by Williams [Will64] and has subsequently been shown to be efficient as well as useful for many applications. A problem with such min ordered (or max ordered) structures arises when a particular application (e.g. external quicksort) requires that both the minimum and maximum keys be found efficiently. An alternative data organization based on the notion of *min-max ordering* has been proposed in [ASSS84]. In a tree that is min-max ordered i) keys on even levels are smaller than or equal to the keys of all descendant nodes; and ii) keys on odd levels are larger than or equal to the keys of all descendant nodes. Thus, the minimum key is located at the root and the maximum key is at one of the children of the root. The min-max ordering strategy when applied to heaps [ASSS84] gives a conceptually simple and efficient double-ended priority queue structure. The structure is called a *min-max heap*. For min-max heaps, algorithms to construct, insert, and delete the minimum or maximum, are similar to those for min heaps.

For min heaps, several lower bounds on the number of comparisons necessary to perform the construct, insert and delete operations have been established [GoMu82]. From an information theoretic view-point both the min-heap as well as the min-max heap have the same information content. For min-max heaps, however, the lower bound is easily strengthened:

Theorem 1. The number of comparisons required in the worst case to construct a min-max heap on a set of n keys is at least $(3n)/2 - 3$ if n is even and $(3(n-1))/2 - 1$ if n is odd.

We now define the convert operation for min-max heaps as follows:

Convert (Q) reorder items in min-max heap Q to obtain a max-min heap and return it. There are at least two reasons for studying this operation. First, we would like to know whether it is possible to find an efficient algorithm for the problem, or phrased differently, ask how distinct is the information in a min-max heap to that contained in a max-min heap on the same set of items. Secondly, conversion is of interest in deriving the lower bound for the merging problem, as we will show later.

A min-max heap of height h is *perfect* if it has exactly $2^{h+1} - 1$ nodes. To find a lower bound for the conversion problem, we define a perfect heap in which all nodes at the last two levels are labelled with distinct keys in the range $(-\infty, +\infty)$. All other nodes

---

[*] This research was supported by the Natural Science and Engineering Council of Canada under Grant No. A0392. (See HaSa[86] for a full version of this paper.)

are labelled with $-\infty$ and $+\infty$, alternating from level to level. We refer to this embedding of keys in a $-\infty,+\infty$ heap as L. Now consider the problem of finding the maximum (or minimum) of all keys at the last two levels in a perfect heap of odd (or even) height labelled by L. Suppose the number of candidates for the maximum key before conversion is x, and after conversion is *at most* y ($x > y$). Then the number of comparisons required for conversion is *at least* $x - y$. By determining x and y, we obtain a lower bound for conversion from which it seems that the information contained in a min-max heap is "qualitatively" different from that contained in a max-min heap:

Theorem 2. The number of comparisons required in the worst case to convert a perfect min-max heap of size $n = 2^{h+1} - 1$ into a max-min heap is at least $(n - 1)/6$ if h is even, and at least $(n + 3)/6$ if h is odd.

The merge operation is defined as follows:

Merge (P, Q) return a min-max heap containing all items in min-max heaps P and Q. A heap is *mergeable* if the merge operation does not require examining a positive fraction of items in the heap. Two recent algorithms show that min heaps of sizes m and n can be merged using O(log m log n) comparisons [SaSt85, StSa85] and are thus mergeable.

The lower bound for the merge operation for min-max heaps is obtained by showing that any conversion problem for min-max heaps is *transformable* to a merging problem. The following lower bound for merging is obtained: The number of comparisons required in the worst case to merge two perfect min-max heaps, each on n/2 keys, into a min-max heap on $n = 2^{h+1} - 1$ keys, is at least n/6 if h is even, and at least $(n + 4)/6$ if h is odd.

Theorem 3. Min-max heaps are not sublinearly mergeable.

Acknowledgements. The authors are grateful to D. Avis, M. D. Atkinson and Th. Strothotte for useful discussions on this topic.

References

[ASSS84] M. D. Atkinson, J.-R. Sack, N. Santoro and Th. Strothotte (July 1984), "An efficient, implicit double-ended priority queue", Technical Report 55, School of Computer Science, Carleton University. Also to appear in *Comm. ACM.*

[GoMu82] G. H. Gonnet and J. I. Munro (July 1982), "Heaps on heaps", *Proc. ICALP* **9**, Aarhus, 282-291.

[HaSa86] A. Hasham and J.-R. Sack (June 1986), "A note on lower bounds for min-max heaps", Technical Report 99, School of Computer Science, Carleton University.

[SaSt85] J.-R. Sack and Th. Strothotte (1985), "An algorithm for merging heaps", *Acta Informatica* **22**, 171-186.

[StSa85] Th. Strothotte and J.-R. Sack (December 1985), "Heaps in heaps", *Congressus Numerantium* **49**, 223-235.

[Will64] J. W. J. Williams (June 1964), "Algorithm 232: Heapsort", *Comm. ACM* **7**:6, 347-348.

GENERATION OF OPTIMAL BINARY SPLIT TREES

J. H. HESTER AND D. S. HIRSCHBERG
Department of Information and Computer Science, University of California
Irvine, California 92717

ABSTRACT

A binary split tree is a search structure combining features of heaps and binary search trees. The fastest previously known algorithm for building an optimal binary split tree required $\Theta(n^5)$ time when access probabilities are distinct, and exponential time otherwise. An algorithm is presented here which improves the time in the case of non-distinct access probabilities to $O(n^5)$. When the smallest access probability of all records is known to be greater than $\Theta(n^{-2})$, the asymptotic time required by this algorithm is further reduced. In the best case, when the smallest access probability is $\Theta(1/n)$, the algorithm requires $O(n^3 \lg n)$ time for distinct access probabilities and $O(n^4 \lg n)$ time for non-distinct access probabilities.

INTRODUCTION

A binary split tree (BST) is a structure for storing records on which searches will be performed, assuming that the probabilities of access are known in advance. For every subtree T in a BST, a record with the largest access probability over all records in T is stored in the root of T. The remaining records are distributed among the left and right subtrees of T such that the keys of all records in the left subtree are less than the keys of all records in the right subtree. Each node in a BST contains the *key value* of the record in that node and a *split value* which lexically divides the values of the keys in the left and right subtrees. A simple split value is the value of the largest key in the left subtree.

Under the assumption of distinct access probabilities and no failed searches, for any given set of n records, the key to be put in the root is predetermined but the split value for the root may be chosen to divide the remaining $n-1$ records between the left and right subtrees in any of n possible ways. If failed searches are considered, the split value may be any of $n+2$ possibilities.

Binary split trees were introduced by Sheil [4], who conjectured that the arbitrary removal of nodes with high access probabilities from the lexicographic ordering (for placement in roots of higher subtrees) made the normal dynamic programming techniques inapplicable. However, Huang and Wong [1] noted that the keys missing from any given range must be the keys with the largest access probabilities in that range of keys, thus allowing a representation of the set of keys in a subtree by specifying a range of keys and a count of the number of keys missing from that range. This led to a $\Theta(n^5)$ time and $\Theta(n^3)$ space algorithm to construct optimal BSTs in a manner similar to Knuth's method [2] for constructing optimal binary search trees. Shortly after, Perl [3] presented a similar but independently derived algorithm which also took into account probabilities of failed searches. Perl also presented an extension which handled non-distinct access probabilities, by including some top-down decision-making, which resulted in an exponential time algorithm.

We describe and analyze an algorithm which reduces the $O(n^5)$ time requirement for distinct access probabilities to $O(n^4)$ and, for non-distinct probabilities, requires only an additional factor of the maximum number of equal probabilities. In the worst case this results in $O(n^5)$ (as opposed to the previous exponential result), and can approach $O(n^4)$ when equal access probabilities are sparse. The asymptotic time required by the algorithm is further reduced whenever the smallest access probability among the records is greater than $\Theta(n^{-2})$. When this is true, a factor of n^2 is reduced to a factor of $n^\alpha \lg(n)$, where the smallest access probability is $\Theta(n^{-\alpha})$. For example, if the smallest access probability is $\Theta(1/n)$, the algorithm requires only $O(n^3 \lg n)$ time for distinct probabilities and $O(n^4 \lg n)$ when there are $\Theta(n)$ non-distinct access probabilities.

BASICS OF THE ALGORITHM

We are given n records indexed from 1 to n. Each record r_i has a key $Key(i)$ such that, for all $i<j$, $Key(i) < Key(j)$. If the records are not so ordered, we can prepend a sort to the algorithm without adversely affecting its asymptotic cost. Each record r_i also has an *access probability* $p(i)$. In addition, to account for failed searches, we are given *failure probabilities* $q(i)$ for $0 \le i \le n$ which are the probabilities of searching for a key K such that $Key(i) < K < Key(i+1)$.

For the following definitions, assume that access probabilities are distinct. The next section gives modifications for enabling the use of non-distinct access probabilities.

Define a *range* of records i to j to be the records whose indices are in the range $i+1, i+2, \ldots, j$. Let $\langle i, j, k\rangle$ refer to the sequence of probabilities

$$\{q(i), p(i+1), q(i+1), p(i+2), \ldots, q(j-1), p(j)\}$$

where the largest k access probabilities (p's) are left out of the sequence. Since records are ordered by key value, the records with the k largest access probabilities could be anywhere in the sequence. A subtree T *spans* the sequence $\langle i, j, k\rangle$ if the subtree contains all records whose access probabilities are in $\langle i, j, k\rangle$, and contains no other records. A record r is said to be *missing* from a subtree T if the index of r is in the range i to j and T spans $\langle i, j, k\rangle$, but r's access probability is not in $\langle i, j, k\rangle$. In other words, r is missing from T if it is in the range of T, but has one of the k highest access probabilities in that range, which causes it to to be placed in the root of some higher subtree.

Perl [3] gave a simple proof that the keys missing from any subtree must be the keys with the largest access probabilities that T spans, and that these keys must be stored in an ancestor of the root of T. This leads to a dynamic programming method for constructing optimal BSTs (for distinct access probabilities) similar to Knuth's method [2] for constructing optimal binary search trees. The primary recurrence is

$$COT[i, j, k] = W[i, j, k] + \min_{i<l<j} \left\{ \begin{array}{c} COT[i, l, GE_L(i, j, k, l)] \\ + \\ COT[l, j, k+1-GE_L(i, j, k, l)] \end{array} \right\}$$

where $COT[i, j, k]$ is the cost of an optimal subtree spanning $\langle i, j, k\rangle$, $W[i, j, k]$ is the total weight of any subtree spanning $\langle i, j, k\rangle$ (which does not include the k missing records), and $GE_L(i, j, k, l)$ is the number of records in the range $\langle i, l, 0\rangle$ which have access probabilities greater than or equal to that of the root of an optimal subtree spanning $\langle i, j, k\rangle$. These values (as well as the outputs mentioned below) may all be calculated in a dynamic programming fashion. The algorithm outputs the arrays $R[i, j, k]$, $SP[i, j, k]$, and $k_L[i, j, k]$, whose values respectively define the index of the root, the split value, and the number of nodes missing, for an optimal subtree spanning $\langle i, j, k\rangle$.

NON-DISTINCT ACCESS PROBABILITIES

The major problem when there may be non-distinct access probabilities is that, during the calculation of COT, SP and k_L, the root of a given subtree may be unknown since it could be any one of a set of non-missing records with maximal access probabilities. It may even be unknown which records are in this set, i.e. which of the records with access probabilities equal to the root are not "already" missing. Perl's algorithm [3] shifted to a top-down approach in this case, which resulted in an exponential time complexity. We note that the only pieces of information needed during the calculation of COT, SP and k_L are the *weights* of the subtrees, and that these weights are not dependent on which one of the records with equal access probabilities is the root or which ones of these records are missing. The only problem is in predicting from which subtree the root, that eventually will be picked, is missing. This is *not* fixed, so we check all possible distributions of potential roots (including missing records which will be roots somewhere above) between the subtrees without ever committing to exactly which record is the root of the current subtree. The final decisions may be made in a top-down fashion when the tree is constructed, at which time only the optimal subtrees are considered and any available record with the right weight is acceptable. Thus the exponential work is avoided.

For the following, let T be any tree spanning $\langle i, j, k\rangle$ and let P be the access probability of any key that might be the root of T. When we compare records, saying that one is greater, less, etc. than another, we are referring to the access probabilities of those records. This also applies when we compare a record to P.

To avoid ambiguity, we refine the definition of $R[i, j, k]$ to be the index of the *rightmost* possible root of T (the record with the greatest key value among the non-missing records with highest access cost). Calculation of the refined R may be performed at no additional cost.

Note that, although the value of $GE_L(i, j, k, l)$ was equal to the number of records missing from the left subtree when only distinct access probabilities were considered, this is no longer true. We replace $GE_L(i, j, k, l)$ with $GT_L(i, j, k, l)$, the number of records in the left subtree which are greater than P, and $EQ_{mL}(i, j, k, l)$, the number of records in the left subtree which are equal to P and are missing from the left subtree of T. Thus the number of missing records in the left subtree is now represented by $GT_L + EQ_{mL}$, which eventually will be stored in $k_L[i,j,k]$ after the optimal value of EQ_{mL} is found.

When looking for optimal splits, we bound the possible values of EQ_{mL} and then check all values within our bounds. This does not determine a root for T, but provides constraints which are used during the final top-down construction of the tree to ensure that the root picked is consistent with the remainder of the tree. For a given range of $\langle i,j,k\rangle$ and a given split index l, define EQ_m as the number of records which are equal to P and are missing from the subtree, EQ_L as the number of records in the range of the left subsubtree (spanning $\langle i,l,0\rangle$) which are equal to P, $EQ_R = EQ - EQ_L$, and $EQ_{mR} = EQ_m + 1 - EQ_{mL}$.

EQ_{mL} may now be bounded in the following manner:

(1) (number of keys missing from left subtree)
$\leq$ (number of keys missing from both subtrees)
(2) $GT_L + EQ_{mL} \leq k+1$ rewriting (1)
(3) $EQ_{mL} \leq EQ_L$
(*) $EQ_{mL} \leq \min\{EQ_L, k+1-GT_L\}$ from (2) and (3)
(4) $EQ_R \geq EQ_{mR}$
(5) $EQ_R \geq EQ_m + 1 - EQ_{mL}$ rewriting (4)
(6) $EQ_{mL} \geq 0$
(**) $EQ_{mL} \geq \max\{0, EQ_m + 1 - EQ_R\}$ from (5) and (6)

Thus, for any l splitting T (and the values of EQ_L and GT_L corresponding to that split), (*) and (**) give us

$$\max\{0, EQ_m + 1 - EQ_R\} \leq EQ_{mL} \leq \min\{EQ_L, k+1-GT_L\}$$

Note that, any time there is only one possible root, these bounds restrict EQ_{mL} to either 1 or 0, depending on whether the single possible root of T is in the left or right subtree of T. Thus, the extra factor of n on the time of this algorithm is only an upper bound; the algorithm is $o(n^5)$ (approaching $\Theta(n^4)$) when there are few records with equal access probabilities, and is $\Theta(n^4)$ when records have distinct access probabilities.

The calculations of R and all arrays of the *GE, GT, EQ*, etc. family, for a given range $\langle i,j,k\rangle$, primarily compare the access probability of record j to the access probabilities of $R[i,j-1,k]$ and $R[i,j-1,k-1]$ in order to determine what role record j plays in the tree spanning $\langle i,j,k\rangle$. In other words, the effects of adding record j to a subtree (which has been previously calculated) consisting of all the records in the range except record j are considered. Determining the appropriate tests requires some thought, but is not particularly esoteric. However, calculation of $W[i,j,k]$ in the presence of non-distinct access probabilities is complicated by the fact that, when a record with index j such that $p(j) = P$ is being considered by the dynamic programming processes, it is sometimes unclear whether that record is present or missing from the subtree. It is simple enough to determine this when $EQ_m(i,j,k) = 0$ but when $EQ_m(i,j,k) > 0$ we do not know which of the records that are equal to P are missing, since the final choice of roots (in higher subtrees) has been postponed. However, we *do* know how many records in the range are equal to P, and thus we may calculate the subtree's weight as the difference of $W\left[i,j,k-EQ_m(i,j,k)\right]$ (the weight of the subtree with *none* of the records equal to P missing, which we know from previous calculation since we work our way up through increasing values of k) and $P{\cdot}EQ_m(i,j,k)$ (the total weight of the records which are equal to P and missing from the subtree for the current value of k).

We construct the tree in a natural top-down fashion based on the values of R, *SP*, k_L and *Key* as before, but the choice of the root for each subtree is made in postorder, after the subtrees below it have been fully constructed. A global array of flags is used to indicate which records have been allocated as roots so far, and the choice of the root for a subtree is restricted to any record in the range of the subtree which has the correct access probability and has not already been allocated as a root of some lower subtree. We search backwards from the rightmost possible root in the range, which may save a little time, but still yields an $O(n)$ search for each root, making the time required to construct the tree (after the arrays have been set up) $O(n^2)$. Thus the total time for the algorithm is dominated by the $O(n^5)$ time required to calculate *COT*, *SP*, and k_L (a factor of $O(n)$ is required for each variable i, j, k, l, and EQ_{mL}).

BOUNDING THE SMALLEST ACCESS PROBABILITY

Intuitively, a large range of records cannot be located too low in an optimal tree due to the fact that the weight of those records, that low in the tree, would make the overall expected cost of the tree too high. We note that any subtree spanning $\langle i, j, k\rangle$ must be at least k deep in the tree, since the k missing nodes must be on the path from the root of the tree to the root of the subtree. We bound the values of k which need be considered for a given range of records by eliminating any values which would result in too great a cost being contributed to the tree.

Sheil [4] defined a median split tree where, by always choosing a split value such that the number of records in the left and right subtrees are equal (or nearly so), an almost complete tree results. Since this is a valid split tree and has a worst case cost of $\lg(n)$ probes, therefore $\lg(n)$ is an upper bound on the expected cost of any optimal split tree. Thus, any subtree whose weight multiplied by its distance from the root is greater than $\lg(n)$ cannot be part of any optimal split tree.

With this in mind, for any given i and j we can determine a range of values of k for which any subtree spanning $\langle i, j, k\rangle$ cannot be a part of an optimal BST, and which therefore need not be considered. Assume that, before beginning the algorithm, we search the list of records to determine s, the value of the smallest access probability among the records. The cost contributed to the entire tree by any subtree spanning $\langle i, j, k\rangle$ is at least $W[i, j, k]\cdot$(the depth of the subtree). Since k is a lower bound on the depth of the subtree and it contains $d - k$ (where $d = j - i$) nodes, each of which weighs at least s, the subtree must contribute a cost of at least $ks(d-k)$ to the split tree. This is greater than $\lg(n)$ for k in the range

$$BADk \quad = \quad \left\{ \frac{d - \sqrt{d^2 - (4/s)\lg n}}{2} < k < \frac{d + \sqrt{d^2 - (4/s)\lg n}}{2} \right\}$$

Thus, values of k in the range $BADk$ need not be considered. Note that this bound is only defined for $d \geq \sqrt{(4/s)\lg n}$. For smaller values of d, $BADk$ is the empty range. We remove the $BADk$ range from the "k" loop of the algorithm. The only other modification that needs to be made is in the innermost loop, when costs of subtrees are looked up in the COT array. A check must be made to see if these subtrees were not considered, (i.e., if $ks(d-k) > \lg n$). If so, their weight was not previously computed, but should be considered to be infinite so as to eliminate possibility of their use.

Several refinements to this technique suggest themselves, using known values instead of the outside limits used above to define the necessary limits on loops. For example, instead of using $\lg(n)$ as the limit on cost for all subtrees, the expected cost of the median split tree for the given data may be calculated and used as a tighter limit. When comparisons are made, the actual weight of the subtree in question is known and may be used instead of the minimum possible value, $s(d-k)$. The lower bound on the expected cost contributed by a subtree can be further refined by noting that, for a subtree at depth (at least) k, there must be k nodes along the path from the root of the tree to the root of the subtree. These nodes all have weight at least as large as the root of the subtree. When calculating the cost contributed by a subtree, the algorithm could include the cost of this path by keeping track of such weighted path lengths, or simply calculating a lower bound on the weight by assuming that all records along the path have a weight equal to that of the subtree's root. Finally, for any subtree which does not span all records, there is a minimum cost that must be contributed by other records. As before, the algorithm could spend a little extra time to get a close evaluation of this cost, or could quickly get a less tight bound by calculating the value based on a few global facts.

The values, in terms of speed improvements, of the refinements above are questionable, since both the potential gain and the added cost appear to be of a non-asymptotic order (without making further assumptions about the data). Thus they will not be considered in the analysis.

ANALYSIS

The analysis is based on the following simple loop outline of the main portion of the algorithm for calculating an optimal BST when there may be non-distinct access probabilities. The interspersed order formulas represent time required by blocks of statements that are of little consequence to the structure of our analysis.

```
Compute()
begin
    $\Theta(n)$
    for $d \leftarrow 2$ until $\sqrt{(4/s)\lg n}$ do
        for $i \leftarrow 0$ until $n+1-d$ do begin
            $j \leftarrow i+d$
            for $k \leftarrow 0$ until $d-1$ do
                Find_Min$(i, j, k)$
            $\Theta(1)$
        end
    for $d \leftarrow \sqrt{(4/s)\lg n}+1$ until $n+1$ do
        for $i \leftarrow 0$ until $n+1-d$ do begin
            $j \leftarrow i+d$
            for $k \leftarrow 0$ until $\left(d-\sqrt{d^2-(4/s)\lg n}\right)/2$ do
                Find_Min$(i, j, k)$
            for $k \leftarrow \left(d+\sqrt{d^2-(4/s)\lg n}\right)/2$ until $d-1$ do
                Find_Min$(i, j, k)$
            $\Theta(1)$
        end
end

Find_Min$(i, j, k)$
begin
    $\Theta(1)$
    for $l \leftarrow i+1$ until $j-1$ do begin
        $\Theta(1)$ or $O(n)$
    end
    $\Theta(1)$
end
```

Figure 1. Loop outline of split tree algorithm.

The innermost block requires $\Theta(1)$ time (to look up the costs of optimal subtrees of a subtree spanning $\langle i, j, k\rangle$ and split at index l) when access probabilities are distinct, and $O(n)$ time otherwise (to consider values of EQ_{mL}, i.e., distributions between left and right subtrees of missing records with access probabilities equal to that of the root). This block will be ignored until near the end of the analysis. The procedures not presented (which precalculate R, W, and the others arrays discussed earlier) contribute no more than $O(n^3)$ time.

Assume that the access probability of the smallest record is $\Theta(n^{-\alpha})$ (i.e., $s = cn^{-\alpha}$ for some constant value c). Substituting this value into the definition of $BADk$ we find that, for values of d greater than $f(n) = \sqrt{(4/c)n^{\alpha}\lg n}$, the number of values of k which can be ignored is at least $\sqrt{d^2-(4/c)n^{\alpha}\lg n}$. Thus the number of values of k which need to be considered in this case is no more than $d-\sqrt{d^2-(4/c)n^{\alpha}\lg n}$. For smaller values of d, the definition of $BADk$ yields no improvement to the old upper bound on the number of values of k which need to be considered, which is d.

Since the number of values of the inner loop variables i, k, and l are all bounded by the value of the outer loop variable d (i will have no more than $n-d$ values, k will be bounded as described above, and l will have no more than d values), an upper bound on the asymptotic time required by the algorithm can be expressed as a sum of the products of these bounds:

$$\begin{aligned}
&\sum_{d=2}^{f(n)}(n-d)\cdot d\cdot d+\sum_{d=f(n)+1}^{n+1}(n-d)\left(d-\sqrt{d^2-(4/c)n^\alpha \lg n}\right)d \\
&< nf^3(n)+n\sum_{d=f(n)+1}^{n+1} d\left(d-\sqrt{d^2-\frac{4n^\alpha \lg n}{c}+\left(\frac{2n^\alpha \lg n}{cd}\right)^2}+\sqrt{\left(\frac{2n^\alpha \lg n}{cd}\right)^2}\right) \\
&\le nf^3(n)+n\sum_{d=1}^{n+1}\tfrac{4}{c}n^\alpha \lg n \\
&= O\left(n^{(3/2)\alpha+1}\lg^{(3/2)} n+n^{\alpha+2}\lg n\right)
\end{aligned}$$

Since the algorithm for distinct access probabilities never takes more than $O(n^4)$ time, $(3/2)\alpha + 1 \le \alpha + 2$ for all $1 \le \alpha \le 2$, and the additional loop to handle non-distinct access probabilities adds at most a factor of n to the asymptotic time, the algorithm has an upper bound of

$$O\left(\min\{n^4, n^{\alpha+2}\lg n\}\right) \qquad \text{for distinct access probabilities}$$
$$O\left(\min\{n^5, n^{\alpha+3}\lg n\}\right) \qquad \text{for non-distinct access probabilities}$$

Thus the algorithm's asymptotic time is decreased whenever $\alpha < 2$.

CONCLUSIONS AND OPEN QUESTIONS

An algorithm has been presented for constructing optimal binary split trees in $O(n^4)$ time when access probabilities are distinct, and $O(n^5)$ time when access probabilities are non-distinct. When the minimum access probability of records is $\Theta(n^{-\alpha})$ (for $\alpha < 2$), a factor of n^2 in the asymptotic time is reduced to a factor of $n^\alpha \lg(n)$. Thus, for example, it is not unreasonable to expect the algorithm to require close to $O(n^3 \lg n)$ time when access probabilities are all proportionally large and there are only a few non-distinct access probabilities.

It seems unlikely that the algorithm we describe requires less time than the upper bound we give, but this is yet to be shown. Thus it is still possible that a tighter time bound may be found by using other expressions for our bound concerning which subtrees may contribute to an optimal tree.

Similarly, using different bounds on possible subtrees may lead to more efficient algorithms.

Further assumptions about the weights of the data may lend value to some or all of the refinements given. Otherwise, only measurement in a given situation can determine if they provide any gain in speed.

REFERENCES

1. S.-H. S. Huang and C. K. Wong, Optimal binary split trees, *J. Algorithms* **5** (1984) 69–79.

2. D. E. Knuth, "The Art of Computer Programming," Vol. 3, "Sorting and Searching," pp. 433–439, Addison–Wesley, Reading, Mass., 1973.

3. Y. Perl, Optimum split trees, *J. Algorithms* **5** (1984) 367–374.

4. B. A. Sheil, Median split trees: A fast lookup technique for frequently occurring keys, *Comm. ACM* **21** (1978) 947–958.

HOW MUCH ON THE AVERAGE IS THE PATRICIA TRIE BETTER?

WOJCIECH SZPANKOWSKI
Department of Computer Science, Purdue University
West Lafayette, IN 47907

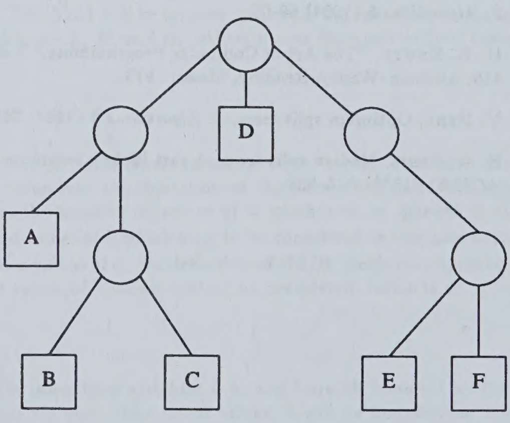
Abstract. *The paper studies the average complexity of the Patricia trie built from n keys over a V-ary (asymmetric) alphabet. We focus on the analysis of the depth of a leaf in the Patricia, and the average as well as the variance of the depth are investigated. In particular, we prove that the variance is* $O(\ln n)$ *for an asymmetric Patricia, and* $O(1)$ *for a symmetric Patricia. In addition, the average number of internal nodes in the trie is studied. These results are compared with the equivalent ones for regular tries.*

1. INTRODUCTION

Digital searching is a well-known technique for storing and retrieving information using lexicographical (digital) structure of words. Let U be an alphabet containing V elements, $U = \{\sigma_1, \dots, \sigma_V\}$ and we define a set S which consists of finite numbers, say n, of (possible infinite) strings (keys) from U. A *trie or radix search trie* is a V-ary digital search tree in which edges are labelled by elements from U and leaves (external nodes) contain the keys [1], [7], [8]. The access path from the root to a leaf is a minimal prefix of the information contained in the leaf. The radix trie (we also call it *regular trie*) has an annoying flaw: there is "one-way branching" which leads to the creation of extra nodes in the tree. D.R. Morrison discovered a way to avoid this problem in a structure which he named *Patricia tree (trie).* In such a tree all nodes have branching degree greater than or equal two. For more details see [7], [8]. Patricia trees find many applications e.g. in lexicographical order, dynamic hashing algorithms and most recently in conflict resolution algorithms (see [1], [3], [7], [8], [11]).

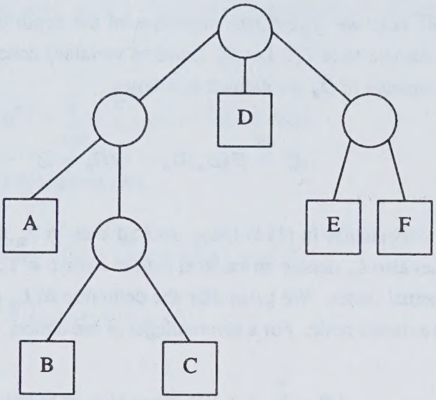
The following is an example of a regular trie built over the alphabet $U = \{0,1,2\}$, that is, V=3.

A = 100
B = 010
C = 012
D = 100
E = 220
F = 221



In the above example the set S of keys is $S = \{A, B, C, D, E, F\}$, hence n=6. The trie consists of two types of nodes, namely, internal nodes and external nodes. The internal nodes are used to determine branching strategy, while keys are stored in the external nodes. Note that in the above example there exists one-way branching on the first level in the third subtree, that is, in the path from the root to the keys E and F. Collapsing this one-way branching of the appropriate internal node we obtain the Patricia trie as shown below.

A = 100
B = 010
C = 012
D = 100
E = 220
F = 221



From the analytical standpoint, direct comparison between the Patricia and regular digital search trie is suggested because they both build search keys into V-ary tree structures, using digital properties of the keys. This paper compares some average complexities of the Patricia and regular tries.

The average complexity analysis of the Patricia is very scarce, and in most investigations the binary Patricia with uniform distribution of keys is studied, that is, for $V = 2$ it is assumed that the probability of occurrence of σ_1 and σ_2 is the same and equal to 0.5. (see [2], [7]). This simplification is dropped in this paper, and we assume that a sequence of elements from U (keys) is an independent sequence of Bernoulli trials. More precisely, if x_k is the k-th element of a key, than the probability $Pr\{x_k = \sigma_i\}$ is equal to p_i, $i = 1 ,\ldots V$. For such a model we show how to compute the average external path length and depth of a leaf. We compare them with regular digital search trie. Moreover, we do not restrict our analysis to the average values of some quantities of interests, as it was done so far. The average value is a very poor measure, and there is a need to study higher moments. We shall show how to obtain the variance of the depth of a leaf. This quantity provides information on *how well a tree is balanced.* For binary symmetric Patricia tries the variance was obtained in [6] (see also [5]). We extend this analysis to asymmetric V-ary Patricia tries. In particular, we show that the variance is $O(\log n)$ for asymmetric Patricia and only $O(1)$ for the symmetric one. This will be compared with an equivalent quantity for a regular digital search trie.

2. PRELIMINARY RESULTS

Let us consider a family T_n of Patricia tries with n keys (records) built over and alphabet $U = \{\sigma_1, \ldots, \sigma_V\}$. A key is a string of (possible infinite) elements from U, such that the i-th element $\sigma_i \in U$ occurs independently of other elements, and with probability p_i, $i = 1, 2, \ldots, V$, $\sum_{i=1}^{V} p_i = 1$. A trie $t \in T_n$ built over U is called V-ary asymmetric Patricia trie since the alphabet contains V elements which are distributed according to the probabilities p_i, $i = 1, 2, \ldots, V$. Two parameters of tries are of particular interest: successful search and unsuccessful search. The successful search, called also the *depth of a leaf*, is the number of internal nodes in the trie on the path from the root to a given key (external node) if the key *is* already in the trie. If the new key *is not* in the trie, then an unsuccessful search occurs. The unsuccessful search is not simply related to the successful search, since unsuccessful searches are more likely to occur at external nodes near the root, and it is not studied in this paper.

In this note we investigate properties of the depth of a leaf (successful search) in a random family of Patricia tries T_n. Let D_n (random variable) denote the successful search in T_n. The m-th factorial moments of D_n are defined as follows

$$d_n^m \overset{def}{=} E\{D_n(D_n - 1)(D_n - 2) \cdots (D_n - m + 1)\}, \tag{1}$$

where the expectation in (1) is taken over all tries in T_n and over all external nodes in a given trie $t \in T_n$. Let also L_n denote an external path in a trie $t \in T_n$, that is, it is the sum of all paths from the root to external nodes. We generalize the definition of L_n as follows. Let $D_n(i)$ be a path from root to the i-th external node. For a given integer m we define

$$L_n^m = \sum_{i=1}^{n} D_n(i)[D_n(i) - 1][D_n(i) - 2] \cdots [D_n(i) - m + 1] \tag{2}$$

and by definition

$$l_n^m \overset{def}{=} EL_n^m \tag{3}$$

The quantity l_n^m is not exactly the m-th factorial moment of L_n, but it is closely related to it. We call l_n^m the m-th semi-factorial moment of the external path length.

It is shown that these moments are related to the m-th derivatives of the so called generating functions of T_n. Let $H_n(z)$ denote this generating function with the coefficient at z^k being the expected number of external nodes at level k in our family of trees. There is no explicit formula for $H_n(z)$ but a rather sophisticated recurrence. To find it, let us denote by $\mathbf{j}=(j_1, j_2, \ldots, j_V)$ a vector such that $j_1 + j_2 + \cdots + j_V = n$. Also let $\left[\begin{matrix} n \\ \mathbf{j} \end{matrix}\right] \overset{def}{=} \left[\begin{matrix} n \\ j_1, \ldots, j_V \end{matrix}\right] = \frac{n!}{j!\, j_2! \cdots j_V!}$ be a multinomial coefficient, and let $\sum_{\{j_\Sigma = n\}} f(j_1, \ldots, j_V)$ denote a sum of $f(j_1, \ldots, j_V)$ over all $\mathbf{j}$ such that $j_1 + j_2 + \cdots + j_V = n$ for a given function $f(\cdot)$. Then the following recurrence on $H_n(z)$ may be established (see [7] and [11]).

Lemma 1. For any natural n the generating function $H_n(z)$ of a random family of Patricia tries, T_n, satisfies the recurrence

$$H_0(z) = 0, \qquad H_1(z) = 1$$
$$H_n(z) = z \sum_{\{j_\Sigma = n\}} \left[\begin{matrix} n \\ \mathbf{j} \end{matrix}\right] p_1^{j_1} \cdots p_V^{j_V} [H_{j_1}(z) + \cdots + H_{j_V}(z)] - (z - 1)[p_1^n + p_2^n + \cdots + p_V^n] \tag{4}$$

□

Denote now by $H_n^{(m)}(1)$ the m-th derivative of $H_n(z)$ at $z = 1$. Then for any integers n and m the following holds (see [7], [10], [11])

$$H_n(1) = n \qquad l_n^m = H_n^{(m)}(1) \tag{5}$$

$$d_n^m = l_n^m / n \tag{6}$$

□

Using Lemma 1 and (5) we derive a recurrence equation for l_n^m, hence by (6) also on d_n^m. In this paper we consider only m=1 and m=2. We shall work at the beginning with l_n^m. For simplicity we perform all computations for $V = 2$ and $p_1 = p$, $p_2 = 1 - p_1 = q$. Then for $m = 1$ we find immediately that

$$l_n^1 = n(1 - p^n - q^n) + \sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k} [l_k^1 + l_{n-k}^1]. \tag{7}$$

Computing the second derivative of $H_n(z)$ one shows that

$$l_n^2 = 2[\sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k} (l_n^1 + l_{n-k}^1) - l_n^1(p^n + q^n)] +$$
$$\sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k} [l_n^2 + l_{n-k}^2].$$

But, the first term of the above is by (7) equal to $l_n^1 - n$, hence

$$l_n^2 = 2(1 - p^n - q^n)[l_n^1 - n] + \sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k} [l_k^2 + l_{n-k}^2]. \tag{8}$$

Note that (7) and (8) is a system of recurrences, i.e., to find l_n^2 we need l_n^1. These recurrences may be easily generalized to obtain the m-th semi-factorial moment of L_n (for details see [11]). Note also that the recurrences have a common pattern and they differ only by the first terms which we call the *additive term* and denote by a_n. This type of recurrence has been solved by Szpankowski in [9]. Since we use it extensively in the further part of the paper we quote below, without proofs, some of these results.

Solution of a recurrence equation

Let $x_0, x_1, \ldots, x_n$ be a sequence of numbers satisfying the following linear recurrence

$$\begin{gathered} \textbf{given} \quad x_0 = x_1 = 0 \\ \textbf{solve} \quad x_n = a_n + \sum_{\{j_\Sigma = n\}} \binom{n}{\mathbf{j}} p_1^{j_1} \cdots p_V^{j_V} [x_{j_1} + \cdots + x_{j_V}] \end{gathered} \tag{9}$$

where a_n is any sequence of numbers. We call a_n an additive term of the recurrence (9). To solve (9) we introduce the so called *binomial inverse relations*. Let us, for a given sequence a_n, define a new sequence $\hat{a}_n$ as

$$\hat{a}_n = \sum_{k=0}^{n} (-1)^k \binom{n}{k} a_k \qquad a_n = \sum_{k=0}^{n} (-1)^k \binom{n}{k} \hat{a}_k \tag{10}$$

(The second equation justify the name binomial inverse relations). For more details see Riordan [12]. Using the above we proved in [9] that

Theorem 1. The recurrence (9) possesses the following solution

$$x_n = \sum_{k=2}^{n} (-1)^k \left[{n \atop k} \right] \frac{\hat{a}_k + k a_1 - a_0}{1 - \sum_{i=1}^{V} p_i^k}. \tag{11}$$

□

From the numerical point of view the solution (11) is not better than the recurrence (9), however, (11) might be used to derive an asymptotic approximation for x_n. In most computer science applications a_n is a linear combination of the following terms $\left[{n \atop r} \right] c^n$, where r is an integer, and c is a constant. From [7], [9] [12], we know that

$$a_n = \left[{n \atop r} \right] c^n \rightarrow \hat{a}_n = \left[{n \atop r} \right] (-c)^r (1-c)^{n-r}. \tag{12}$$

Since the recurrence (9) is linear, from the asymptotic point of view we need an approximation for large n of the following

$$T_{n,r}(c) \overset{def}{=} \sum_{k=2}^{n} (-1)^k \left[{n \atop k} \right] \left[{k \atop r} \right] \frac{c^k}{1 - \sum_{i=1}^{V} p_i^k}. \tag{13}$$

Let $h_k = (-1)^k \sum_{i=1}^{V} p_i \, ln^k \, p_i$. Then in [9] we have proved that

Theorem 2. For any r, c and large n the following holds

$$T_{n,r}(c) = \begin{cases} nc\{\dfrac{ln(nc) + \gamma - \delta_{n,0}}{h_1} + \dfrac{h_2}{2h_1^2} + (-1)^r f_r(nc)\} + O(1) & r = 0,1 \\ \\ (-1)^r \; nc\{\dfrac{1}{r(r-1)h_1} + f_r(nc)\} + O(1) & r \geq 2 \end{cases} \tag{14}$$

where $\gamma = 0.571$ is the Euler constant, and $f_r(n)$ is a fluctuating function with a small amplitude defined as

$$f_r(n) = - \sum_{\{z_k^r \neq 0,\, r-1\}} \frac{\Gamma(z_k^r) n^{r - z_k^r}}{\sum_{i=1}^{V} p_i^{r+1-z_k^r} \ln p_i}. \tag{15}$$

The number z_k^r, $k = 0, 1, \ldots,$ $r = 0, 1, \ldots,$ n are roots of the equation

$$1 - \sum_{i=1}^{V} p_i^{r-z} = 0. \tag{16}$$

□

It is shown [2], [3], [7] [9] that the function $f_r(n)$ has a very small amplitude and may be safely ignored in practice.

3. MAIN RESULTS

In this section we derive asymptotic approximation for the average depth of a leaf, the variance of the depth and the average number of internal nodes.

The average external path length

We have shown before that the average external path length, l_n^1, satisfies recurrence (7). But (7) falls into our general recurrence (9) with $a_n = n\,(1-p^n - q^n)$. Hence using (12) and Theorem 1 we immediately obtain the following solution for l_n^1

$$l_n^1 = \sum_{k=2}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix} \begin{bmatrix} k \\ 1 \end{bmatrix} \frac{pq^{k-1} + qp^{k-1}}{1-p^k - q^k} \tag{17}$$

To find asymptotic approximation for (17) note that by (13) we can express (17) as $l_n^1 = p/q\, T_{n,1}(p) + q/p\, T_{n,1}(q)$. Hence, by Theorem 2

$$l_n^1 = n\{ \frac{ln\ n}{h_1} - \frac{\bar{h}_1 - \gamma}{h_1} + \frac{h_2}{2h_1} + F_1(n)\} + O(1) \tag{18}$$

where $\bar{h}_1 \overset{def}{=} -p\ln(1-p) - q\ln(1-q)$, and $F_1(n)$ is a fluctuating function with very small amplitude, and may be safely ignored in practice.

Let d_n^T and d_n^P denote the depth of a leaf (l_n/n) for a regular digital trie and the Patricia one respectively. Then using (18) and the result from [10] we show that

$$d_n^T - d_n^P = \frac{\bar{h}_1}{h_1} + O(n^{-1}) \tag{19}$$

The above holds for any V-ary asymmetric Patricia trie if one defines h_1 and $\bar{h}_1$ as follows: $h_1 = \sum_{i=1}^{V} p_i \ln p_i$ and $\bar{h}_1 = \sum_{i=1}^{V} p_i \ln(1-p_i)$. For symmetric binary case $d_n^T - d_n^P = 1$.

The variance of the external path length

The computation of the variance is much more intricate. Note first that

$$\begin{aligned} \text{var}\, L_n &= l_n^2 + l_n^1 - \frac{1}{n}(l_n^1)^2 \\ \sigma_n^2 = \text{var}\, D_n &= \frac{\text{var}\, L_n}{n} \end{aligned} \tag{20}$$

hence knowing l_n^2 we are able to compute var D_n. We now estimate l_n^2. Note that by (8) l_n^2 satisfies the recurrence (9), but now $a_n = (1 - p^n - q^n)(l_n^1 - n)$. Hence, the inverse sequence $\hat{a}_n$ is (we use (12))

$$\hat{a}_n = 2\hat{l}_n^1 + 2(\delta_{n,1} - npq^{n-1} - npq^{n-1}) - 2I(p^n l_n^1) - 2I(q^n l_n^1). \tag{21}$$

where $I(p^n\, l_n^1)$ is the inverse relation to $p^n\, l_n^1$. Two problems are encountered in obtaining the exact expression for $\hat{a}_n$. Namely, we must know $\hat{l}_n^1$, and - what is more intricate - the inverse sequence to $p^n l_n^1$. The first problem is easy. We know that l_n^1 is given by (17). But by the definition of the inverse relations $l_n^1 = \sum_{k=0}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix} \hat{l}_k^1$ and $\hat{l}_0 = \hat{l}_1 = 0$. Hence combining this with (17) we immediately find

$$\hat{l}_k^1 = k\, \frac{p\,q^{k-1} + q\,p^{k-1}}{1 - p^k - q^k}. \tag{22}$$

In [11] we proved that $I(p^n\, l_n^1)$ is given by

$$I(p^n l_n^1) = \sum_{j=0}^{n} \begin{bmatrix} n \\ j \end{bmatrix} \begin{bmatrix} j \\ 1 \end{bmatrix} p^j (1-p)^{n-j} \frac{pq^{j-1} + qp^{j-1}}{1 - p^j - q^j} \tag{23}$$

Using the above, after some algebra, we finally obtain the following formula for l_n^2

$$\begin{aligned} l_n^2 = {} & 2 \sum_{k=1}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix} \begin{bmatrix} k \\ 1 \end{bmatrix} \frac{(p^k + q^k)(pq^{k-1} + qp^{k-1})}{(1 - p^k - q^k)^2} - \\ & 2 \sum_{k=2}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix} \frac{1}{1 - p^k - q^k} \sum_{j=2}^{k} \begin{bmatrix} k \\ j \end{bmatrix} \frac{(p^j q^{k-j} + q^j p^{k-j})(pq^{j-1} - qp^{j-1})}{1 - p^j - q^j} \end{aligned} \tag{24}$$

Hereafter we deal with the asymptotic approximation for the variance, var D_n, of the depth of a leaf. For this we need an asymptotic expression for (24). Let us denote by A_n and B_n the first and the second term in (24), that is, $l_n^2 = 2A_n - 2B_n$.

The expression for A_n does not fall into (13) (since the denominator is raised to the power two), and hence Theorem 2 cannot be applied. However, for asymptotic approximation of A_n we need to evaluate the following

$$T_n^{(2)}(c) = \sum_{k=2}^{n} (-1)^k \begin{bmatrix} n \\ k \end{bmatrix} \begin{bmatrix} k \\ 1 \end{bmatrix} \frac{c^k}{(1 - \sum_{i=1}^{V} p_i^k)^2} \tag{25}$$

which generalized (13). We use the Mellin transform technique [4], [10], [11]. Performing some algebra over (25) we can prove that (for details see [11])

$$T_n^{(2)}(c) = -\frac{nc}{2\pi i} \int_{-\frac{1}{2}-i\infty}^{-\frac{1}{2}+i\infty} \frac{\Gamma(z)(nc)^{-z}}{(1 - \sum_{i=1}^{V} p_i^{1-z})^2}\, dz + O(1) \tag{26}$$

where $\Gamma(z)$ is the gamma function [4]. The evaluation of the integral is standard: we must compute residues of the function under the integral to the right of the line $(½-i\,\infty,\ -½+i\,\infty)$. Let z_k $k=0,\pm1,\pm2,\ldots$ denote roots of the denominator, that is,

$$1-\sum_{i=1}^{V} p_i^{1-z}=0. \tag{27}$$

Note that $z_0=0$, while the other roots are complex. For more detailed treatment of (27) see [2], [5], [7]. The most difficult to handle is the pole $z_0=0$ since it is the double pole of the denominator and, in addition, singular point of the gamma function. It is also proved that this pole gives the biggest contribution to the asymptotic approximation. Using the Taylor expansion of the functions under the integral in (26) we can prove that the residue at $z_0=0$ is

$$\frac{1}{2h_1^2}\ln^2 nc+\varepsilon\ln nc+\delta$$

where

$$\begin{aligned}\varepsilon&=\frac{\gamma h_1+h_2}{h_1^3}\\ \delta&=\frac{1}{h_1^2}\left[\frac{\pi^2}{12}+\frac{\gamma^2}{2}+\frac{3h_2^2}{4h_1^2}+\frac{\gamma h_2}{h_1}-\frac{h_3}{3h_1}\right].\end{aligned} \tag{28}$$

So finally after some algebra we obtain

$$T_n^{(2)}=\frac{1}{2h_1^2}\ln^2 nc+\varepsilon\ln nc+\delta+f^{(2)}(nc)+O(1) \tag{29}$$

where $f^{(2)}(n)$ is a fluctuating function with a small amplitude. Note now that by (24) and (26)

$$A_n=\sum_{i=1}^{2}\frac{p_i}{1-p_i}\sum_{j=1}^{2}T_n^{(2)}[(p_j(1-p_i)] \tag{30}$$

hence

$$A_n=n\left\{\frac{1}{2h_1^2}\ln^2 n+\left[\varepsilon-\frac{h_1+\bar{h}_1}{h_1^2}\right]\ln n+\eta+F_A(n)\right\}+O(1) \tag{31}$$

where

$$\eta=\frac{1}{2h_1^2}(h_2+\bar{h}_2+2h_1\bar{h}_1)-\varepsilon(h_1+\bar{h}_1)+\delta \tag{32}$$

and $F_A(n)$ is a fluctuating function with a small amplitude, which might be safely ignored in practice.

The analysis of B_n is much more intricate, however, we can reduce B_n to a form where only $T_{n,1}(p_i)$ appear. Details of the computations are omitted and the reader should consult [11]. Concluding , we have shown in this section that (see [11])

PROPOSITION.

(i) The mean of the successful search D_n for large n is given by

$$ED_n=\frac{1}{h_1}\{\ln n+\rho+F_1(n)\}+O(n^{-1}) \tag{33}$$

where $\rho \overset{def}{=} \gamma - \bar{h}_1 + \frac{h_2}{2h_1}$ and $F_1(n)$ is a fluctuating function with a small amplitude.

(ii) The variance, var D_n, of the successful search for large n satisfies

$$\operatorname{var} D_n = \frac{h_2 - h_1^2}{h_1^3} \, ln \; n + \alpha - 2\beta + F_2(n) + O(n^{-1}), \tag{34}$$

where

$$\begin{gathered}\alpha = \frac{1}{h_1^2}\left[\frac{\pi^2}{6} + \gamma^2 + \frac{3}{2}\frac{h_2^2}{h_1^3} + \frac{2\gamma h_2}{h_1} - \frac{2}{3}\frac{h_3}{h_1} + h_2 + \bar{h}_2 + 2h_1\bar{h}_1\right] - \\ 2(h_1 + \bar{h}_1)\frac{\gamma h_1 + h_2}{h_1^3} + \frac{\rho}{h_1}\left(1 - \frac{\rho}{h_1}\right)\end{gathered} \tag{35}$$

and

$$\beta = \frac{1}{h_1}\sum_{\lambda=1}^{V}\sum_{\upsilon=1}^{V} p_\upsilon p_\lambda \sum_{l=0}^{\infty}\sum_{\{i_\Sigma = l\}} \binom{l}{\mathbf{i}} \prod_{\mu=1}^{V} p_\mu^{i_\mu} \ln\{1 + \frac{p_\lambda(1 - p_\upsilon)}{1 - p_\lambda}\prod_{\mu=1}^{V} p_\mu^{i_\mu}\}. \tag{36}$$

In particularly, for V-ary symmetric Patricia tries $h_2 = h_1^2$ and (34) is reduced to (the coefficient at ln n is equal to zero)

$$\operatorname{var} D_n = \frac{\pi^2}{6\ln^2 V} + \frac{1}{12} - \frac{2}{\ln V}\ln\{\prod_{l=1}^{\infty}(1 + \frac{1}{V^l})\} + F_2(n) + O(n^{-1}). \tag{37}$$

□

We now compare the successful search for regular tries and Patricia tries. Let $D_n^{[T]}$ and $D_n^{[P]}$ denote the depth of a leaf for the regular trie and the Patricia trie respectively. The variance of D_n for a regular trie is given by (34) with $\beta = 0$ [10]. In the table below we show the variances for tries and the Patricia tries in the symmetric case. For $V = 2$ (symmetric case) the result was previously obtained by Kirschenhofer and Prodinger [6] (see also [10]).

V	var$D_n^{[T]}$	var$D_n^{[P]}$
2	3.507	1.000
3	1.446	0.630
4	0.939	0.500
5	0.718	0.430
6	0.596	0.387

Finally, let us mention that the recurrence for the variance is very slowly "convergent" to its asymptotic approximation (at least 150 terms must be computed), so the above results are particularly useful.

The average number of internal nodes

It is clear that binary asymmetric Patricia has $n - 1$ internal nodes. It might be also easily proved that regular tries have on the average n/h_1 internal nodes. For symmetric case it gives the difference between the number of internal nodes for both tries equal to $n\ (1-\ln 2)/\ln 2$. The situation is a little more complicated when the V-ary Patricia trie is analyzed. In that case, the average number of internal nodes, I_n, satisfies a recurrence equation which is of the same type as (9) with $a_n = 1-\sum_{i=1}^{V} p_i^n$. Then using (12) and Theorem 1 one immediately proves that

$$I_n=\sum_{k=2}^{n}(-1)^k\binom{n}{k}\frac{V-1-\sum_{i=1}^{V}(1-p_i)^k}{1-\sum_{i=1}^{V}p_i^k} \tag{38}$$

Hence, by (13) and Theorem 2 $I_n=(V-1)T_{n,0}(1)-\sum_{i=1}^{V}T_{n,0}(1-p_i)$. Simple computations reveal that defining $\underline{h}_1=\sum_{i=1}^{V}(1-p_i)\ln(1-p_i)$ one finds

$$I_n = n\ [\underline{h}_1/h_1+f(n)]-1/(V-1)+O(1) \tag{39}$$

For V=2 (39) reduces to $I_n=n-1$ as desired. For general V-ary tries the difference between the average number of internal nodes in a regular trie, I_n^T, and the Patricia, I_n^P is $I_n^T-I_n^P=n\ (1-\underline{h}_1/h_1)$.

REFERENCES

[1] Aho, A., Hopcroft, J., Ullman, J., Data structures and algorithms, Addison-Wesley, 1983.

[2] Flajolet, Ph., Sedgewick, R., Digital search trees revisited, *SIAM J. Comput.*, 15, 1986, pp.748-767.

[3] Gonnet, G., Handbook of algorithms and data structure, Addison-Wesley, 1984.

[4] Henrici, P., Applied and computational complex analysis, vol.I & II, John Wiley&Sons, New York 1977.

[5] Jacquet, Ph., and Regnier, M., Trie partitioning process: Limiting distributions, *Proc. of CAAP'86*, Nice 1986, pp.196-210.

[6] Kirschenhofer, P., Prodinger, H., Some further results on digital trees, *ICALP 86*, to appear.

[7] Knuth, D., The art of computer programming, Addison-Wesley, 1973.

[8] Sedgewick, R., Algorithms, Addison-Wesley, 1983.

[9] Szpankowski, W., Analysis of a recurrence equation arising in stack-type algorithms for collision-detecting channels, *Proc. Intern. Seminar on Computer Networking and Performance Evaluation*, Tokyo 1985, 399–412.

[10] Szpankowski, W., Some results on V-ary asymmetric tries, *Purdue University*, CSD-TR 582, October 1985.

[11] Szpankowski, W., Patricia tries again revisited, *Purdue University*, CSD-TR 625, 1986.

[12] Riordan, J. Combinatorial identities, John Wiley&Sons, 1968.

ON RECOGNIZING POLYGONS, OR HOW TO EAVESDROP

JAMES A. DEAN[1], ANDRZEJ LINGAS[2] and
JÖRG-R. SACK[3]

Abstract

For the task of spying on a closed room (given as a simple polygon P) we are equipped with a camera and a microphone. The equipment is strong enough to pick up signals which have penetrated through at most one wall. The task is to be able to receive any signal which originates inside the room. One problem is to determine whether there exist a placement of the audio/video equipment so that we may hear/see the room through its exterior walls. If such a location exists then P is pseudo star-shaped. Another, more general, problem is to determine all possible such locations. We give algorithms for testing a given n-vertex polygon P for pseudo star-shapedness from a given point, in linear time, and for determining all possible placements for the equipment in O(nr) time, where r is the number of convex vertices of P.

The class of pseudo star-shaped polygons generalizes, and the notion of pseudo kernel unifies, the common classes of convex, star-shaped and monotone polygons. Many standard computational geometry problems such as triangulation, point-inclusion etc. can be solved as efficiently for pseudo star-shaped polygons as for the more restricted polygon classes.

1. Introduction

In two dimensions objects are typically represented as polygons. For many applications (e.g. pattern recognition) structural knowledge such as e.g. convexity is available about the specific objects under investigation. Therefore, depending on the application area, different polygon classes have been introduced and efficient algorithms solving geometric problems have been developed for these polygon classes. Standard polygon classes are convex, star-shaped or monotone polygons [see e.g. PS85].

---

[1] Bell-Northern Research, Ottawa, Canada; research for this paper was done while the author was at Carleton University.

[2] Department of Computer and Information Science, University of Linköping, Linköping, Sweden; research for this paper was done while the author was visiting Carleton University.

[3] School of Computer Science, Carleton University, Ottawa, Canada. This research was in part supported by NSERC under grant No.A0392 and by Carleton University.

Star-shaped polygons are not necessarily monotone, and vice versa. However, as we will see in Section 3, all three polygon classes are related through a particular visibility problem, we referred to as the *Spy Problem.* The spy problem is informally described as follows: To do some undercover surveillance on a closed room, we are equipped with an infra-red camera and/or a microphone which can pick up signals which have penetrated through at most one wall. If a single location suffices to do surveillance on the entire room, the polygon representing the room is referred to as *pseudo star-shaped.* A formal definition of pseudo star-shapedness is given in Section 2.

Two spy-problems naturally arise and their solutions are discussed in Section 4:
(1) Given a specific location for the equipment can we hear/see the entire room from that position?
(2) Where, (outside or inside) the room, can we place the surveillance equipment to see/hear the entire room?
The answers to the spy-problems can obviously be found by testing for pseudo star-shapedness either from a given point, or for pseudo star-shapedness when no point is given, respectively, where the latter problem seems to be the harder.

Any convex, star-shaped, or monotone polygon can be shown to be pseudo star-shaped; however, there are pseudo star-shaped polygons which do not fall into any of these classes. In Section 3, we will further examine these relationships and provide necessary and sufficient conditions for a pseudo star-shaped polygon to be convex, star-shaped, or monotone. Although the class of pseudo star-shaped polygons is more general than these traditional computational geometry classes the main structural property allowing efficient algorithms' design is maintained. We will illustrate this on some examples in Section 5.

2. Preliminaries

We define the class of pseudo star-shaped polygons (a restricted version of which was introduced in Ke85]). To see how this class is related to problems in computer graphics we introduce some terminology on visibility. Let $P = (p_1,\dots,p_n)$ be a simple polygon. A *polygonal chain* $C_{i,j} = p_i,\dots, p_j$ of P is *visible from a point u* if there exists a point on the chain that is visible from u. A polygonal chain is *completely visible from a point u* if each point on the chain is visible from u. We also allow the point u to be located at infinity in this case the visibility of the *perspective model* , as used up to now, becomes that of the *parallel model,* i.e. that in which a chain is completely visible from u at infinity if all points on the chain are visible by a set of rays originating at u and parallel to each other [see e.g. NS78]. Each different orientation of rays determines a different point at infinity.

We now define pseudo star-shaped polygons in the following way:
Let u be a point in the plane or at infinity. A polygon $P=(p_0,\dots, p_{n-1})$ is *pseudo star-shaped from a given point u* if the boundary of P can be partitioned into two polygonal chains $C_{i,j}$ and $C_{j,i}$ such that each point on $C_{i,j}$ is visible from u, even in

the presence of $C_{j,i}$; and $C_{j,i}$ is completely visible from u, in the absence of $C_{i,j}$. A polygon $P=(p_1,..., p_n)$ is *pseudo star-shaped* if there exists a point u from which P is pseudo star-shaped. (See Figure1 for an example.) This definition is quaivalent to the informal definition given above. The set of points from which P is pseudo star-shaped is called the *pseudo kernel* of P. The vertices p_i and p_j are called the *defining vertices* for chains $C_{i,j}$ and $C_{j,i}$.

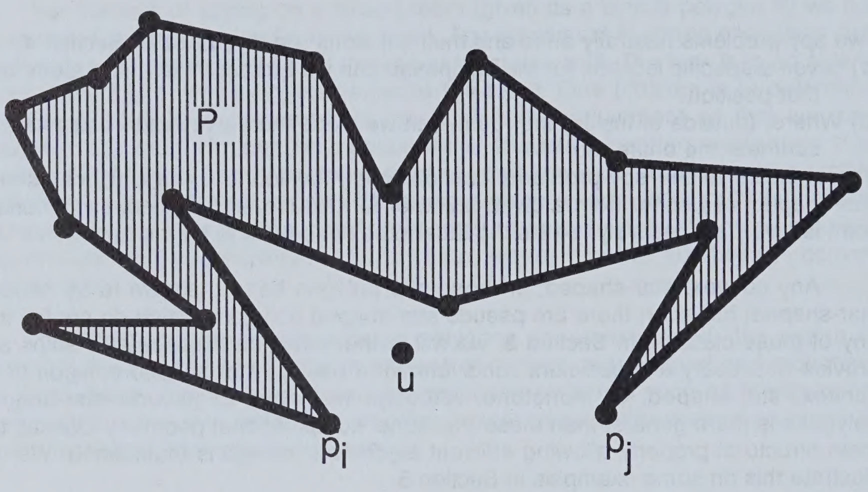


Figure 1: A polygon pseudo star-shaped from the point u.

3. Properties of Pseudo Star-Shaped Polygons

In this section, we derive some properties of pseudo star-shaped polygons and establish the relationship of this polygon class to the three standard polygon classes consisting of convex, star-shaped, or monotone polygons.

Kernels and Pseudo Kernels

In the case of star-shaped polygons the set of all points from which P is star-shaped is connected, convex and lies inside P. The pseudo kernel of P, however, may lie outside P, be non convex, and might even be disconnected. In the latter case, it is convenient to say that P has several pseudo kernels. In this way there is one connected, convex pseudo kernel, for each pair of chain-defining vertices.

A pseudo kernel is called an **outer pseudo kernel** if it lies outside P and an **inner pseudo kernel**, otherwise. Outer pseudo kernels may be polygonally **bounded** or may be **unbounded**, i.e. have "edges" extending to infinity. Figure 2 illustrates a pseudo star-shaped polygon with one of its pseudo kernels unbounded.

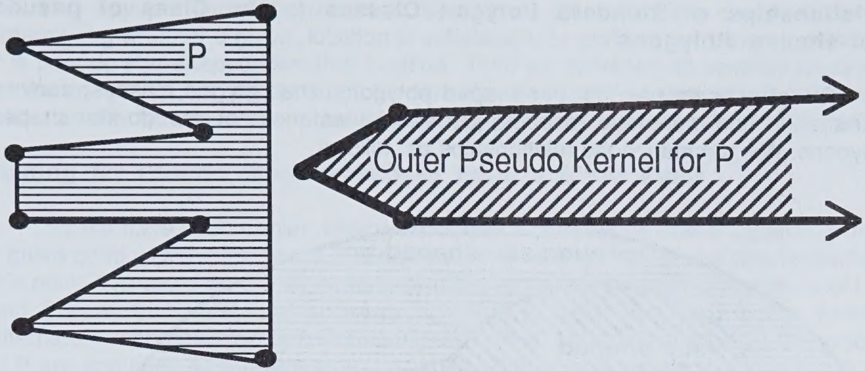


Figure 2: A pseudo star-shaped polygon and one of its outer kernels.

In the next Lemma we give a characterization of pseudo star-shaped polygon which might serve as an alternate definition of pseudo star-shaped polygons and which will be utilized in later sections.

Lemma 2.1 *Let u be a point in the plane. A polygon P is pseudo star-shaped from u iff there exist indices i,j such that both* $P_{ij} = (p_i ,..., p_j ,\ u)$ *and* $P_{ji} = (p_j ,..., p_i ,\ u)$ *are star-shaped from u.*

Proof: See the full version of this paper [DLS86] for this and other proofs here omited.

Theorem 2.2 *Kernel(P) = Inner Pseudo Kernel(P)*

Relationship to Monotone Polygons

Unbounded pseudo kernels, OK(P), may be classified according to which points of the plane they contain (in addition to the points at infinity).

(1) OK(P) contains no point on the plane.
(2) OK(P) contains all points on the plane.
(3) OK(P) contains some subset of the plane.

In case (2) P is monotone to all directions. In case (3) OK(P) can be described by some polygonal chain and half-lines. Let $OK_i(P)$ be an outer pseudo kernel of a polygon pseudo star-shaped P. Two edges of $OK_i(P)$, say e_a and e_b defining the unbounded edge of the outer pseudo kernel extend to infinity. Lines perpendicular to these rays define directions of monotonicity.Thus unbounded kernels define directions of monotonicity.

Preparata and Suppowit [PS81] observed that an n-vertex polygon may be monotone to as many as O(n) disjoint regions of monotonicity and thus there may exist at least O(n) unbounded disjoint outer kernels.

Theorem 2.3 *A polygon P is pseudo star-shaped from u at infinity iff P is monotone in the direction perpendicular to the visibility rays originating at u.*

Relationships of Standard Polygon Classes to the Class of pseudo star-shaped Polygons

By introducing pseudo star-shaped polygons, the polygon classes: convex, star-shaped and monotone become just special instances of pseudo star-shaped polygons. We summarize the relationships as follows.

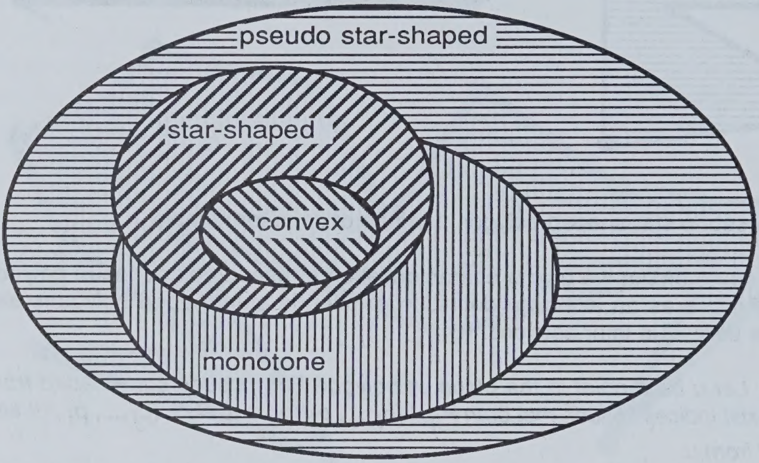


Figure 3: Illustrating the set containment relationship between polygon classes.

Polygon Class	Pseudo Kernel
convex	entire plane including infinity
star-shaped	non-empty inner pseudo kernel
monotone	unbounded outer pseudo kernel
pseudo star-shaped	non-empty pseudo kernel

Figure 4: Characterization of polygon classes via pseudo kernels.

4. Solutions to the Spy Problems

Recall that for the task of spying on a closed room (given as a simple polygon P) we are equipped with a camera and a microphone. The equipment is strong enough to pick up signals which have penetrated through at most one wall. The task is to be able to receive any signal which originates inside the room. Of the two resulting spy problems discussed in Section 1, we first solve the problem of determining whether a given location is satisfactory to perform our task, i.e. whether P is pseudo star-shaped from that location. Then we construct all possible locations for the equipment, i.e. we construct the pseudo kernel for P. A desirable location can subsequently be chosen from these possible locations subject to other conditions.

Testing for Pseudo Star-shapedness From a Given Point

As we have seen earlier, testing a polygon P for pseudo star-shapedness from a given point u in the interior of P is equivalent to testing P for star-shapedness from this point. We could do this by constructing the kernel of P using the algorithm of Lee and Preparata [LP79] and subsequently testing u for inclusion in the kernel. Alternately, without explicitly constructing the kernel, we can test whether all vertices of P are angularly sorted around u, i.e. whether their polar angles are sorted, where the polar angles are taken with u as origin. Both approaches require linear time.

When u is outside P the situation is more interesting. We show that for any given location u outside a simple n-vertex polygon $P=(p_1,\ldots,p_n)$ there is at most one pair of chain-defining vertices. Depending on the location of u there may not be any such pair of vertices. We show how to find the chain-defining vertices and how to determine their existence. We distinguish between u outside the convex hull CH(P) and u in CH(P)\P (where "\" denotes the set difference). Let H(u,x) denote the infinite half-line originating at u through point x (where x may also be located at infinity); notice that H(u,x) is directed. A vertex p_i of P is called a *support vertex* of P with respect to u if no point in P except boundary points are located on $H(u,p_i)$.

Lemma 4.1 *Every pair of chain-defining vertices is a pair of support vertices.*

Notice that for a point u outside CH(P) a support vertex will always exist, whereas this is not necessarily true for points inside CH(P)\P. The following describes how to determine if a support vertex for a point u exists and how it can be found. If u is outside CH(P) then support vertices are easily found, see [PS85]. Otherwise, we compute the region of P that is visible from u. This is obtained, in linear-time, from the solution to the hidden-line elimination problem for polygon P with view point u [see e.g. Le83 or DS85]. If the solution is a (closed) polygon no support vertex can exist. Otherwise, the solution is a non-closed polygonal chain $C_{i,j}$. The endpoints, p_i and p_j, of this chain are support vertices. Once the support vertices p_i, p_j have been found we verify whether the support vertices are chain-defining. This is done by testing whether the vertices on each of the two chains are angularly sorted around u.

Lemma 4.2 *Given a point u outside P then the chain-defining vertices for u can be found in linear time.*

For some applications support vertices for points must be found repeatedly. Using a technique developed in [ST85], this task can be performed in O(log n) time per query, where n is the number of vertices of P.

Theorem 4.3 *In linear-time a given polygon can be efficiently tested for pseudo star-shapedness from a given point.*

Constructing Pseudo Kernels

In this section, we shall address the problem of constructing the pseudo kernel of a simple polygon. By Theorem 2.2 , the inner pseudo kernel of a simple polygon is equal to the kernel of the polygon. Since the latter can be constructed by the algorithm of Lee and Preparata in linear time, the most difficult part of the construction of the pseudo kernel of a polygon seems to be the construction of its outer part.

We approach the problem of constructing the pseudo kernel of a simple polygon $P = (p_0, p_1, \dots, p_{n-1})$ by considering for each pair i, j , where $0 \leq i,j \leq n-1$ and $i \neq j$, the set PK(i,j) of all points in the outer pseudo kernel of P such that p_i and p_j are the support vertices for u and u can see the chain $C_{j,i}$ even in the presence of the chain $C_{i,j}$. By convention, we assume that $C_{j,i} = P$ and $C_{i,j} = \emptyset$ if i=j. Thus, for any $0 \leq i \leq n-1$, the set PK(i,i) is the inner pseudo kernel of P. Clearly, the pseudo kernel of P is the union of PK(i,j), where $0 \leq i,j \leq n-1$ and $i \neq j$ or i=0 and j=0. Denote by e_i the edge of P directed from $p_{i-1 \bmod n}$ to p_i (in this section, modulo operations on indices are taken only when specified). Next, let $LP(e_j)$ and $RP(e_j)$ respectively denote the left and the right half-plane induced by e_j. The following lemma yields an obvious way of constructing the sets PK(i,j).

Lemma 4.4 *Given a simple polygon $P=(p_0, p_1, \dots, p_{n-1})$, for $0 \leq i,j \leq n-1$ and $i \neq j$ or i=0 and j=0, the set PK(i,j) is the common intersection of $LP(e_{i+1 \bmod n})$, $LP(e_{i+2 \bmod n})$,...,$LP(e_{j \bmod n})$ with $RP(e_{j+1 \bmod n})$, $RP(e_{j+2 \bmod n})$,...,$RP(e_i)$.*

Suppose that the intersection of the half-planes from Lemma 4.4 can be found in linear time. Let r be the number of convex (from inside) vertices of P. Then, by separately constructing each of the regions PK(i,j) where p_i and p_j are different convex vertices of P, or i=0 and j=0, we could construct the pseudo kernel of P in time $O(nr^2)$. Note that a convex polygon with n vertices has $\Omega(n^2)$ non-empty regions PK(i,j). Therefore, any algorithm for constructing the pseudo kernel of P which constructs all the non-empty regions PK(i,j) takes $\Omega(n^2)$ time in the worst case. We shall show how to construct all the regions PK(i,j) and the pseudo kernel of P in time O(nr).

Of course, it is possible that the problems of constructing the pseudo kernel of P or especially of determining whether a polygon has a non-empty outer pseudo kernel can be solved faster, in subquadratic time. Because of space considerations, we shall omit several not difficult refinements of the O(nr)-time solution to the construction problem, which may speed up the solution when the number of non-empty outer pseudo kernels is subquadratic. The solution we shall outline reduces the original problem for P to r subproblems of the following form:

> *Given a convex vertex p_k of P, construct PK(k,j) for all convex vertices v_j of P where $j \neq k$.*

To obtain the O(nr)-time solution, we need to solve the subproblem in linear time. We may assume without loss of generality that k=0. For j=1,...,n, let OK(j) denote the common intersection of $LP(e_1)$, $LP(e_2)$,...,$LP(e_{j \bmod n})$, and for j=0,1,...,n-1, let IK(j) denote the common intersection of $RP(e_{j+1 \bmod n})$, $RP(e_{j+2 \bmod n})$,...,$RP(e_0)$. Note that OK(n) is the kernel of P. In a natural way, we can extend the above definition by letting OK(0) and IK(n) stand for the plane, and that IK(0) = ∅. Then, using Lemma 4.4 it holds:

i) for j=0,..,n-1, OK(j+1) is a convex subset of $LP(e_{j+1 \bmod n}) \cap OK(j)$,
ii) for j=1,2,...,n-1, IK(j) is a convex subset of $RP(e_{j+1 \bmod n}) \cap IK(j+1)$,
iii) for j=1,2,..,n, $OK(j) \cap IK(j) = PK(0, j \bmod n)$.

Actually, the linear-time algorithm of Lee and Preparata for finding the kernel of a polygon [LP79] applied to P explicitly constructs the sequence OK(1), OK(2),...,OK(n). The region OK(j+1) is constructed on the basis of OK(j) by finding a polygonal chain of the form $w''w_{s-1}...w_tw'$ or $w''w_s...w_{t-1}w'$, where w' and w" are points on the straight-line co-linear with e_j or symbolize a half straight-line starting from the adjacent vertex in the chain, and cutting the chain off OK(j) along the straight-line (see pp. 417,418 in [LP79]). Since OK(j) is convex, the cut off polygonal region is also convex. We can interpret the cut off region as the set difference, od(j), between OK(j) and OK(j+1), for j=1,..,n-1. Similarly, for j=0,...,n-1, let us define, id(j+1), as the set difference between IK(j) and IK(j+1). In a natural way, we can extend the above definitions assuming that od(n) is the inner kernel of P, i.e. it equals OK(n).

By (i), (ii), the above definition and the correctness and the linear-time running time of the algorithm of Lee and Preparata [LP79], we obtain the following, properties of the sets od(0), od(1), od(2),..., od(n):

a) od(j) is a convex subset of $RP(e_{j+1 \bmod n})$,
b) the total number of vertices of od(0), od(1), od(2),..., od(n) is O(n),
c) all the polygonal regions od(0), od(1), od(2),..., od(n) can be constructed in time O(n).

Using the symmetry between the definition of OK(j) , od(j) and that of IK(j), id(j), we can easily modify the algorithm of Lee and Preparata to construct the sequences IK(n-1),...,IK(1) and id(n), id(n-1),...,id(1) in linear time. The modification consists in respectively exchanging the adjectives "left" and "right" in the body of the algorithm. In consequence, we obtain the following, symmetrical list of properties of the polygonal regions id(1),...,id(n):

d) id(j) is a convex subset of $LP(e_{j+1 \bmod n})$,
e) the total number of vertices of id(1),...,id(n) is O(n),
f) all the polygonal regions id(1),...,id(n) can be constructed in time O(n).

In the context of the properties (iii), (b) and (e), the following lemma explains the usefulness of the polygonal regions od(j) and id(j) in finding outer pseudo kernels of P.

Lemma 4.5 For j=1,...,n, it holds that $OK(j) \cap IK(j) = od(j) \cap id(j)$.

By (iii) and the above lemma, given the sequences od(0), od(1),..., od(n-1) and id(1),..., id(n-1), to solve the subproblem for the vertex v_0 it is sufficient to find the intersections $OK(j) \cap IK(j) = od(j) \cap id(j)$ for j=1,...,n-1. Thus, the whole algorithm for constructing the pseudo kernel of a simple polygon can be outlined as follows.

Algorithm I *Construct Pseudo Kernel of a Simple Polygon*
input: a simple polygon $Q=(w_0, w_1,...,w_{n-1})$
output: all non-empty regions PK(i,j) for Q and the pseudo kernel of Q

```
for each convex vertex $w_k$ of Q do
   begin
      rename the vertices of Q such that $( w_k, w_{k+1 \bmod n},...,w_{k+n-1 \bmod n}) =$
         $( p_0, p_1,..., p_{n-1})$;
      find the sequences od(0), od(1),..., od(n) and id(1),...,id(n);
      for j=1,...,n-1 do
      if $w_{k+j \bmod n}$ is convex then
         begin
            PK(k,k+j mod n) := od(j mod n) $\cap$ id(j);
            return PK(k,k+j mod n)
         end
   end
PK(0,0) := od(n);
return PK(0,0);
return the union of PK(0,0) with PK(i,j), where $w_i$ and $w_j$ are convex vertices of Q
and $i \neq j$
```

Theorem 4.6 *Algorithm I constructs all outer pseudo kernels of a simple polygon with r convex vertices and n-r concave vertices in time O(nr).*

5. Geometrical Algorithms for Pseudo Star-shaped Polygons

We have seen how star-shapedness and monotonicity are related to pseudo star-shapedness; monotonicity implies a linear ordering of the vertices on the two chains, while star-shapedness implies an angular ordering. Pseudo star-shapedness encompasses both these orderings. In making use of this more general ordering relation, several other problems studied in computational geometry, like point inclusion, triangulation, union and intersection problems, can be solved efficiently. The interested reader is referred to [De85 and DL86] for further details.

Creating Pseudo Star-Shaped Polygons

Given a set of n points a pseudo star-shaped polygon on these points can be created in O(n log n) time. This is accomplished by first choosing a point u from which the polygon will be pseudo star-shaped. Then the two chains are constructed. The details are left as an exercise to the reader. It is pointed out that the choice of u will determine which polygon type is created. Thus we get a method of generating different polygon classes.

Final Remarks

We have presented a new class of polygons which unifies and generalizes three well-known polygon classes (convex, star-shaped and monotone polygons). Two algorithms for detecting pseudo star-shapedness have been developed. It seems that to improve the O(nr) time algorithm for determining whether a polygon is pseudo star-shaped it is necessary to find out new properties of star-shaped polygons. Some algorithms and applications for pseudo star-shaped polygons have been given.

It is known in Graphics that parallel and perspective projections and thus parallel and perspective visibility are similar and therefore can be transformed from one into the other [see NeS79]. Further possible application areas of the ideas presented here may thus lie in the field of computer graphics.

References

[De85] Dean, J.A., "Structural properties of polygons", Master's thesis, Carleton University, April 1985.

[DL86] Dean, J.A., A. Lingas and J.-R. Sack, "Recognizing polygons: or how to eavesdrop", Tech. Rept. Tr-SCS-99, Carleton University, Ottawa, July (1986).

[DS85] Dean, J.A. and J.-R. Sack, "Efficient hidden-line elimination by capturing winding information", *Proceedings of the 23rd Allerton Conference on Communication, Control and Computing,* Urbana-Champaign, pp. 496-505 (1985).

[Ke85] Keil, J.M., "Decomposing a polygon into simpler components", *SIAM Journal of Computing,* Vol. 14, pp. 799-817 (1985).

[Le83] Lee, D.T., "Visibility of a simple polygon", *Computer Vision, Graphics and Image Processing* Vol. 22, pp. 207-221 (1983).

[LP79] Lee, D.T. and F.P. Preparata, "An optimal algorithm for finding the kernel of a simple polygon", *Journal of the ACM,* Vol. 26, no.3, pp. 415-421 (1979).

[NeS78] Newman, W. and R. Sproull, *Principles of interactive computer graphics,* McGraw-Hill, New York, (1978).

[PS81] Preparata, F.P. and K.J. Supowit, "Testing a simple polygon for monotonicity", *Information Processing Letters,* Vol. 12, pp. 161-164 (1981).

[PS85] Preparata, F.P. and M.I. Shamos, *Computational Geometry,* Springer Verlag, Heidelberg, New York, (1985).

[ST85] Sack, J.-R. and G.T. Toussaint, "Translating polygons in the plane", *Proceedings STACS85,* Saarbrücken, Lecture Notes in Computer Science, pp. 310-321, (1985).

IMPROVED BOUNDS FOR RECTANGULAR AND GUILLOTINE PARTITIONS‡

TEOFILO F. GONZALEZ and SI-QING ZHENG
Department of Computer Science
University of California
Santa Barbara, CA 93106

ABSTRACT

We study the problem of partitioning a rectangle R with a set of interior points Q into rectangles by introducing a set of line segments of least total length. The set of partitioning line segments must include every point in Q. Since this problem is computationally intractable (NP-hard), several approximation algorithms for its solution have been developed. In this paper we show that the length of an optimal guillotine partition is not greater than 1.75 the length of an optimal rectangular partition. Since an optimal guillotine partition can be obtained quickly, we have an efficient approximation algorithm for finding near-optimal rectangular partitions.

INTRODUCTION

Given a rectangular boundary S and a set Q of points inside S, we study the problem of partitioning S into rectangles in such a way that every point in Q lies on at least one of the partitioning line segments and the total length of the partitioning line segments is least possible. Such a partition is called an ***optimal rectangular partition.*** Lingas ***et. al.*** [8] show that finding an optimal rectangular partition is a computationally intractable problem (NP-hard). Since then several approximation algorithms have been proposed, i.e., algorithms for which $\mathrm{LEN}(E_{apx}(\mathrm{I})) \leq c * \mathrm{LEN}(E_{opt}(\mathrm{I}))$, where $E_{apx}(\mathrm{I})$ is the set of partitioning line segments given by the approximation algorithm, $E_{opt}(\mathrm{I})$ is the set of partitioning line segments in an optimal solution, c is some constant, and LEN(E(I)) is the sum of the length of the partitioning line segments in E(I). Gonzalez and Zheng [4] present a divide-and-conquer approximation algorithm that generates solutions such that $\mathrm{LEN}(E_{apx}(\mathrm{I})) \leq 3+\sqrt{3} * \mathrm{LEN}(E_{opt}(\mathrm{I}))$. The time complexity for their algorithm is $O(\ n^2\ )$, where n is the number of points in set Q. Recently, Levcopoulos [10] showed that it is possible to implement this approximation algorithm in O(n log n) time. In [5] an (n^4) approximation algorithm that guarantees solutions such that $\mathrm{LEN}(E_{apx}(\mathrm{I})) \leq 3 * \mathrm{LEN}(E_{opt}(\mathrm{I}))$. The approximation bound is smaller than the one in [4], however there is a substantial difference between the time complexity of these two algorithms. The algorithm in [5] consists of two steps. In the first step, the original problem is transformed into a simpler polynomial time solvable optimization problem; in the second step an existing $O(n^4)$ algorithm is employed to solve the new optimization problem. In this paper we present an approximation algorithm that generates solutions whose objective function value is within 75% of the objective function value of an optimal solution. Gonzalez and Zheng [5] show how to modify the algorithm to generate approximation solutions to a more general version of the problem (bounded boundary problem). The approximation bound obtained for the new problem in [5] is smaller than the one in [7]. If instead of a rectangle we start with a rectilinear polygon and if instead of interior points, the polygon contains holes (a hole is a rectilinear polygon without interior holes), the problem of finding a good rectangular partition becomes much more complex. This problem has applications in computer-aided design of integrated circuits and system for dividing routing regions into channels [11]. Several approximation algorithms for this problem exist (see [1], [7], [9] and [10]). The algorithms with the smallest worst case approximation bound are the ones given in [9] and [10]. The algorithm given in [10] uses as a subalgorithm the procedure given in [4]. Since the approximation algorithm given in this paper generates solutions of the same form as the ones in [4] but closer to the optimal solution value, we conjecture that a smaller approximation bound can be obtained by using the algorithm in this paper.

---

‡ This research was supported in part by the National Science Foundation under Grant DCR - 8503163

We say that the rectangular partition E for I = (S,Q) has a *guillotine cut* if there is a line segment in E that partitions S into two rectangles. We say that a rectangular partition E for I = (S,Q) is a *guillotine partition* if either E is empty (note that Q must be empty) or E has a guillotine cut that partitions S into S_1 and S_2, and both E_1 (partitioning line segments from E in S_1) and E_2 (partitioning line segments from E in S_2) are guillotine partitions for $I_1 = (S_1, Q_1)$ and $I_2 = (S_2, Q_2)$, respectively. An *optimal guillotine partition* is a guillotine partition whose partitioning line segments have least total length. It is simple to see that any guillotine partition is a rectangular partition, but the converse is not true. An optimal guillotine partition can be found in O(n^5) time [12]. Du *et. al.* [2] show that the length of an optimal guillotine partition is no more than twice the length of an optimal rectangular partition. Therefore, an approximation algorithm for the rectangular partition problem is reduced to the problem of finding an optimal guillotine partition. Gonzalez, Shing and Zheng [3] present a simple proof for this fact and point out that it is unlikely that the dynamic programming algorithm can be speeded up. The algorithm in [4] always generates a guillotine partition, however this in not true for the algorithm given in [5].

For problem instance I = (S, Q), let E_{ogp}(I) be the set of partitioning line segments in an optimal guillotine partition and let E_{opt}(I) be the set of partitioning line segments in any optimal rectangular partition. In what follows we show that LEN(E_{ogp}(I)) $\leq$ 1.75 * LEN(E_{opt}(I)). Therefore, we have an O(n^5) approximation algorithm for the rectangular partitioning problem such that LEN(E_{apx}(I)) $\leq$ 1.75 * LEN(E_{opt}(I)).

BOUNDS FOR GUILLOTINE PARTITIONS

We use P to denote the tuple (I = (S , Q) , E(I)), where I is a problem instance and E(I) is any rectangular partition for I. We present a transformation that introduces a set of line segments E'(I) such that E'(I) $\cup$ E(I) forms a guillotine partition (of course E(I) $\cap$ E'(I) = $\emptyset$). The transformation is performed in such a way that LEN(E'(I) $\cup$ E(I)) $\leq$ 1.75 * LEN(E(I)). Applying this transformation to any optimal rectangular partition E_{opt}(I), we know that the resulting guillotine partition E'(I) $\cup$ E_{opt}(I) is such that LEN(E'(I) $\cup$ E_{opt}(I)) $\geq$ LEN(E_{ogp}(I)) since E_{ogp}(I) is an optimal guillotine partition.

Let E_v(I) and E_h(I) represent the sets of vertical and horizontal line segments in E(I), respectively. For a vertical (horizontal) line segment *l*, we use x(*l*) (y(*l*)) to denote the x-coordinate (y-coordinate) of *l* and use B(*l*) (L(*l*)) and T(*l*) (R(*l*)) to denote the y-coordinate (x-coordinate) of the lower (left) end point and the upper (right) end point of the vertical (horizontal) line segment *l*, respectively. The y-coordinate of the bottom and top side of S is given by B(S) and T(S), respectively. The x-coordinate of the left and right side of S is given by L(S) and R(S), respectively. Let X = R(S) - L(S) and let Y = T(S) - B(S).

Since rotation of P by 90 degrees generates an equivalent problem, we may assume without loss of generality that LEN(E_v(I)) $\leq$ LEN(E_h(I)). In what follows we claim that our transformation process introduces a set of vertical line segments E_v'(I) such that LEN(E_v'(I)) $\leq$ LEN(E_v(I)), and a set of horizontal line segments E_h'(I) such that LEN(E_h'(I)) $\leq$ 0.5 * LEN(E_h(I)). Therefore, LEN(E'(I) $\cup$ E(I)) = LEN(E_h'(I)) + LEN(E_v'(I)) + LEN(E(I)) $\leq$ 0.5 * LEN(E_h(I)) + LEN(E_v(I)) + LEN(E(I)) $\leq$ 0.5 * LEN(E_v(I)) + 1.5 * LEN(E(I)) $\leq$ 1.75 * LEN(E(I)).

We say that line segment *l* is *covered* by line segment *l*' if every point in *l* is in *l*'. The line segment *l* is said to be covered by E(I) if there is a line segment *l*' in E(I) such that *l* is covered by *l*'. We use the (corrupted) notation *l* $\in$ *l*' and *l* $\in$ E(I) to indicate line covering. The *overlap* of line segments *l* and *l*' is defined as the line segment *l* $\cap$ *l*'. The overlap of two sets of line segments is defined similarly. We say that a line segment *l* $\in$ E(I) is a vertical (horizontal) *guillotine cut* of S if T(*l*) = T(S) (L(*l*) = L(S)) and B(*l*) = B(S) (R(*l*) = R(S)). Note that this definition is equivalent to the one for guillotine cuts introduced in the previous section. A line segment *l* is a vertical (horizontal) *full cut* of S if T(*l*) = T(S) (L(*l*) = L(S)) and B(*l*) = B(S) (R(*l*) = R(S)). Note that every guillotine cut is a full cut, but not every full cut is a guillotine cut, simply because a full cut is not required to be in E(I). Clearly, when there is a guillotine cut *l* of

S in E(I), P is partitioned into P_1 and P_2 without introducing any new line segment. At this point we (recursively) transform $E_1(I_1)$ and $E_2(I_2)$. If at each step of this recursive transformation we encounter an instance with a guillotine cut, then $E_v'(I) = E_h'(I) = \emptyset$ and our claim for the 1.75 bound follows. However, when there is no guillotine cut of S in E(I) we must introduce a full cut. Selecting the full cut is the crucial part of the transformation.

When there is no guillotine cut of S in E(I) we either introduce a vertical full cut or a set of vertical and horizontal full cuts, depending on the configuration of E(I). The concept of separability, as we shall see later, plays an important role in this decision. We say that a vertical full cut l is *left (right) covered by* $E_v(I)$ if for every point p in l there exists a line segment $l' \in E_v(I)$ such that $x(l') \leq x(l)$ ($x(l') \geq x(l)$) and $B(l') \leq y(p) \leq T(l')$. A *vertical through cut* is a vertical full cut that is both left and right covered by $E_v(I)$. We say P is *vertically separable* if there is at least one vertical through cut in P.

At this time it is convenient to view our transformation as follows:

```
case
  :$E(I) = \emptyset$: return;
  :E(I) has a guillotine cut:
          partition I along a guillotine cut and recursively transform the
                resulting subproblems;
  :E(I) has no guillotine cut but it is vertically separable:
          partition I along a vertical through cut that overlaps with at least one vertical
                line segment in E(I) and recursively transform the resulting subproblems;
endcase
```

If at each step of the transformation process just described we encounter an instance with no internal line segments or an instance with a guillotine cut or an instance that it is vertically separable, then since the transformation process introduces no new horizontal line segments, we know that $E_h'(I) = \emptyset$. The set $E_v'(I) \neq \emptyset$ if in the recursive process we encounter a nonempty problem instance without a guillotine cut. In lemma 1 we prove that for this case LEN($E_v'(I)$) $\leq$ LEN($E_v(I)$). We use the projection function, p($E_v(I)$), to project onto the right side of S the set of vertical line segments $E_v(I)$. The length of the projection is given by LEN(p($E_v(I)$)). Clearly, for every $E_v(I)$, LEN(p($E_v(I)$)) ≥ 0. Instead of proving that LEN($E_v'(I)$) $\leq$ LEN($E_v(I)$), we prove a stronger result, i.e., LEN($E_v'(I)$) + LEN(p($E_v(I)$)) $\leq$ LEN($E_v(I)$).

Lemma 1: For every P = (I,E(I)) our transformation process introduces a set of line segments $E_v'(I)$ such that, LEN($E_v'(I)$) + LEN(p($E_v(I)$)) $\leq$ LEN($E_v(I)$).
Proof: For brevity the proof is omitted. The proof appears in [6].

□

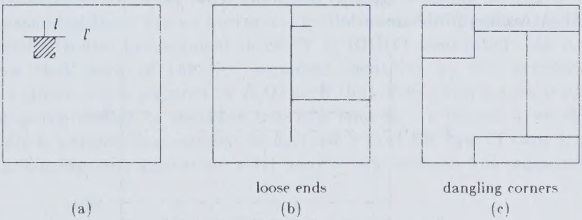
For problem instances P with the properties mentioned above, we know that our claim for the 1.75 bound is correct. For any general problem instance P we cannot yet claim this bound. This is because our transformation process is not complete, there are nonempty and not separable problem instances without a guillotine cut. For those cases we shall apply a three-phase transformation to be carried out by procedure HVH_CUT. In the first phase of procedure HVH_CUT, we introduce a set of horizontal full cuts to partition P into a set of vertically separable subinstances. Let H(I) be the set of horizontal full cuts introduced in phase one. Let $H_1(I) = H(I) \cap E_h(I)$ and let $H_1'(I) = H(I) - H_1(I)$. At this point it is impossible to prove that LEN($H_1'(I)$) $\leq$ 0.5 LEN($H_1(I)$). This is why we need to perform the following steps. In the second phase, each of the vertically separable problem instances constructed in phase one is partitioned by introducing a vertical through cut. The vertical through cuts are carefully selected so that in the next phase we can find a set of horizontal guillotine cuts. Let $H_3(I)$ be this set of horizontal guillotine cuts. Note that $H_3(I) \subseteq E_h(I)$. At this point we claim that LEN($H_1'(I)$) $\leq$ 0.5(LEN($H_1(I)$) + LEN($H_3(I)$)). Our transformation process is formally defined below.

```
procedure TRANS( P = ( I = ( S , Q ) , E(I) ) )
    case
      :E(I) is empty: return;
      :P has a guillotine cut $l$:
          partition P ( along $l$ ) into $P_1$ and $P_2$;
          recursively apply TRANS to $P_1$ and $P_2$;
      :P is vertically separable:
          let $l$ be any vertical through cut in P that overlaps with at least one vertical line segment in E(I);
          partition P ( along $l$ ) into $P_1$ and $P_2$;
          recursively apply TRANS to $P_1$ and $P_2$;
      :else: use procedure HVH_CUT to partition P into $P_1$, ..., $P_u$;
             recursively apply TRANS to each $P_i$;
    endcase
end of procedure TRANS
```

From procedure TRANS and our informal description of procedure HVH_CUT, we know that every vertical line segment introduced is a vertical through cut. Therefore, a proof similar to the one for lemma 1 can be used to show that LEN(E_v'(I)) $\leq$ LEN(E_v(I)). To prove our 1.75 bound it is only required to show that for every P we invoke procedure HVH_CUT, LEN(H_1'(I)) ≤ 0.5 * (LEN(H_1(I)) + LEN(H_3(I))). Hereafter we concentrate on nonempty problem instances, P, which do not have a guillotine cut and are not separable.

The proof for the above bound is not simple. Before proving it we need to introduce some additional notation and prove some intermediate results.

We say that P' = (I' = (S', Q') , E(I')) $\subseteq$ P = (I = (S, Q) , E(I)) if T(S') $\leq$ T(S), B(S') $\geq$ B(S), L(S') $\geq$ L(S), R(S') $\leq$ R(S), Q' contains all the points in Q located inside (not in the boundary) S', and E(I') contains all the line segments in E(I) located inside S'. We say that P' = (I' = (S',Q') , E(I')) is empty if E(I') is empty, i.e., there are no line segments in E(I'). An important property of empty subproblems is given by the following lemma.



loose ends

dangling corners

(a)

(b)

(c)

Figure 2.1:

Lemma 2: If there is an empty subproblem P' = (I',E(I')) $\subseteq$ P = (I,E(I)) and there is a line segment $l \in E_v$(I) such that L(S') < x(l) < R(S') and B(l) = T(S') (T(l) = B(S')), then there is a horizontal line segment $l' \in E_h$(I) such that y(l') = T(S') (y(l') = B(S')), L(l') $\leq$ L(S') and R(l') $\geq$ R(S').

Proof: For brevity the proof is omitted. The proof appears in [6].

□

Among all vertical through cuts in a vertically separable problem P the one with smallest x-coordinate and the one with largest x-coordinate are referred to as the *leftmost vertical through cut lm(P)* and the *rightmost vertical through cut rm(P),* respectively. Note that for some P, the leftmost vertical cut could also be the rightmost vertical cut. In what follows we define some separable subproblems (via procedure MARK), then examine some of their properties and finally show how to use these subproblems and their properties to perform the three phases in procedure HVH_CUT.

Let $y_1 < y_2 <, ..., < y_s$ be the distinct y-coordinates of the set of line segments in $E_h(\mathrm{I})$. Let $y_0 = \mathrm{B(S)}$ and $y_{s+1} = \mathrm{T(S)}$. For $0 \leq t \leq u \leq s+1$, let $S_{t,u}$ denote the rectangle defined by ((L(S), y_t), (L(S), y_u), (R(S), y_u), (R(S), y_t)). Similarly, let $P_{t,u}$ denote P restricted to $S_{t,u}$. It is easy to see that if $P_{t,u}$ and $P_{u,v}$ are vertically separable but $P_{t,v}$ is not vertically separable, then either $\mathrm{x}(\mathrm{lm}(P_{t,u})) > \mathrm{x}(\mathrm{rm}(P_{u,v}))$ or $\mathrm{x}(\mathrm{rm}(P_{t,u})) < \mathrm{x}(\mathrm{lm}(P_{u,v}))$. In the former case we call the $P_{t,v}$ an *LR-decreasing problem* and in the latter case $P_{t,v}$ is referred to as an *LR-increasing problem*. If $P_{i,j}$ is separable then $P_{h,g}$, where $i \leq h \leq g \leq j$, is also separable. Furthermore $\mathrm{x}(\mathrm{lm}(P_{h,g})) \leq \mathrm{x}(\mathrm{lm}(P_{i,j}))$ and $\mathrm{x}(\mathrm{rm}(P_{h,g})) \geq \mathrm{x}(\mathrm{rm}(P_{i,j}))$. Note that $P_{i,i}$ is separable for all i; and for problem instances without guillotine cuts, $P_{i,i+1}$ is always separable.

Procedure MARK

```
    j ← 0;
    for i ← 1 to s+1 do
        if $P_{j,i}$ is not separable then { assign the label A to $y_{i-1}$; j ← i - 1 }
    endfor
    i ← s+1;
    for j ← s to 0 by -1 do
        if $P_{j,i}$ is not separable then { assign the label B to $y_{j+1}$; i ← j + 1 }
    endfor
end of MARK
```

Let $y_{i(1)} < y_{i(2)} < ... < y_{i(k)}$, be the y-coordinates labeled A and let $y_{j(1)} < y_{j(2)} < ... < y_{j(k')}$, be the y-coordinates labeled B by algorithm MARK. Note that the $y_{j()}$'s appear in reverse order from the way procedure MARK labels them. For convenience, let $y_{i(0)} = y_{j(0)} = \mathrm{B(S)}$ and $y_{i(k+1)} = y_{j(k'+1)} = \mathrm{T(S)}$. Since the $P_{i(p),i(p+1)}$ is separable and the algorithm selects the $y_{j()}$'s to represent maximal separable subproblems with respect to the previous $y_{j()}$'s, we know that it is impossible for two $y_{j()}$'s to be in the interval $(y_{i(p)} , y_{i(p+1)}]$. Similarly, it is impossible for two $y_{i()}$'s to be in the interval $[y_{j(p)} , y_{j(p+1)})$. Hence, k = k' and $y_{j(0)} = y_{i(0)} < y_{j(1)} \leq y_{i(1)} < y_{j(2)} \leq y_{i(2)} < ... < y_{j(k)} \leq y_{i(k)} < y_{j(k+1)} = y_{i(k+1)}$. In lemma 3 we prove an important property of LR-decreasing and LR-increasing problems.

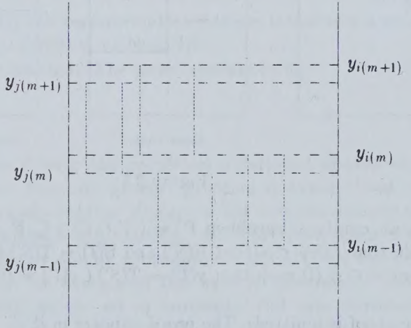


Figure 2.2:

Lemma 3: If $P_{i(m-1),i(m+1)}$ is an LR-decreasing (LR-increasing) problem then $P_{j(m-1),j(m+1)}$ is also an LR-decreasing (LR-increasing) problem.
Proof: For brevity the proof is omitted. The proof appears in [6].

□

Let LEFT = { l | $l \in E_h(\mathrm{I})$ and $\mathrm{L}(l) = \mathrm{L(S)}$} and RIGHT = { l | $l \in E_h(\mathrm{I})$ and $\mathrm{R}(l) =$

R(S)}. In the following lemma we show that for each $y_{i()}$ and $y_{j()}$, there is a distinct horizontal line segment from LEFT or RIGHT associated with it. For $x_1 < x_2$ we use *HLS(y,x_1,x_2)* to represent the horizontal line segment with end points (x_1,y) and (x_2,y).

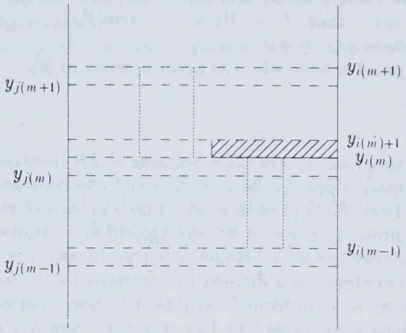


Figure 2.3:

Lemma 4:
(i) If $P_{i(m-1).i(m+1)}$ is an LR-decreasing (LR-increasing) problem then the line segment
l = HLS($y_{i(m)}$, x(rm($P_{i(m).i(m)+1}$)), R(S)) (HLS($y_{i(m)}$, L(S), x(lm($P_{i(m).i(m)+1}$)))) is in E_h(I); and
(ii) if $P_{j(m-1).j(m+1)}$ is an LR-decreasing (LR-increasing) problem then then the line segment
l = HLS($y_{j(m)}$, L(S), x(lm($P_{j(m)-1.j(m)}$))) (HLS($y_{j(m)}$, x(rm($P_{j(m)-1.j(m)}$)), R(S))) is in E_h(I).
Proof: For brevity the proof is omitted. The proof appears in [6].

□

The line segment identified by lemma 4(i) is referred to by $l_{i(m)}$ and the one identified by lemma 4(ii) is referred to by $l_{j(m)}$. From lemma 3 and 4 we know that $l_{i(m)} \in$ LEFT (RIGHT) iff $l_{j(m)} \in$ RIGHT (LEFT). If y($l_{i(m)}$) = y($l_{j(m)}$), then it is not possible for $l_{i(m)}$ and $l_{j(m)}$ to overlap because we are assuming there are no horizontal guillotine cuts. Therefore, for each $y_{i()}$ and $y_{j()}$, there is a distinct horizontal line segment in LEFT or RIGHT associated with it. In the next two lemmas we show that each of the line segments identified by the previous lemma can be associated with a distinct line segment in E_h(I) such that their total length is at least X. This is an important property needed to establish our 1.75 bounds. In lemma 5 we show that for each $l_{i(m)}$ and $l_{j(m)}$ there is another line segment in E_h(I) such that the sum of their length is at least X. Since this does not necessarily guarantee a 1-1 association between line segments, we need lemma 6.

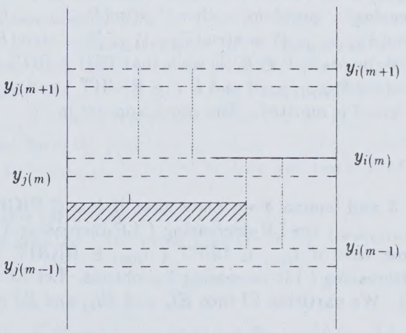


Figure 2.4:

Lemma 5:
(i) If $P_{i(m-1),i(m+1)}$ is an LR-decreasing (LR-increasing) problem then there exists at least one line segment l in E_h(I) such that l = HLS(y, L(S), x(lm($P_{i(m-1),j(m)}$))) (HLS(y, x(rm($P_{i(m-1),j(m)}$)), R(S))), where $y_{i(m-1)} < y \leq y_{j(m)}$; and
(ii) if $P_{j(m-1),j(m+1)}$ is an LR-decreasing (LR-increasing) problem then there exists at least one line segment l in E_h(I) such that l = HLS(y, x(rm($P_{i(m),j(m+1)}$)), R(S)) (HLS(y, L(S), x(lm($P_{i(m),j(m+1)}$)))), where $y_{i(m)} \leq y < y_{j(m+1)}$.
Proof: For brevity the proof is omitted. The proof appears in [6].

□

When all the subproblems are LR-decreasing or LR-increasing, the previous lemma would suffice for our transformation process, because it would associate each segment $l_{i(m)}$ and $l_{j(m)}$ with a unique line segment from E_h(I) in such a way that the sum of their length is greater than X (this is a fundamental property required by our algorithm). However, in general, there are LR-decreasing problems interleaved with LR-increasing problems. For this case the previous lemma does not guarantee the existence of a distinct line segment that could be associated with each $l_{i(m)}$ and $l_{j(m)}$. That is why we need to identify at least two line segments in some regions. Note that in general not all regions have these two line segments, however the two line segments always exist when there is an LR-decreasing problem interleaved with an LR-increasing problem (or vice versa).

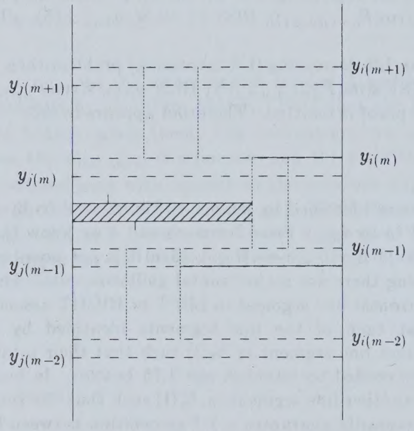


Figure 2.5:

Lemma 6: If $P_{i(m-1),i(m+1)}$ is an LR-decreasing (LR-increasing) problem and $P_{i(m-2),i(m)}$ is an LR-increasing (LR-decreasing) problem, then x(lm($P_{i(m-1),j(m)}$)) = x(lm($P_{j(m-1),j(m)}$)) = x(lm($P_{i(m-1),i(m)}$)) (x(rm($P_{i(m-1),j(m)}$)) = x(rm($P_{j(m-1),j(m)}$)) = x(rm($P_{i(m-1),i(m)}$))) and there are at least two distinct line segments l, l' ∈ E_h(I) such that R(l) = R(l') = x(lm($P_{i(m-1),j(m)}$)) and l, l' ∈ LEFT (L(l) = L(l') = x(rm($P_{i(m-1),j(m)}$)) and l, l' ∈ RIGHT) and $y_{i(m-1)} \leq$ y(l) < y(l') $\leq y_{j(m)}$.
Proof: For brevity the proof is omitted. The proof appears in [6].

□

From lemma 3 and lemma 4 we know that if $l_{i(m)}$ ∈ RIGHT ($l_{i(m)}$ ∈ LEFT) then both $P_{i(m-1),i(m+1)}$ and $P_{j(m-1),j(m+1)}$ are LR-decreasing (LR-increasing) problems. Also from lemma 3 and lemma 4 we know that if $l_{j(m)}$ ∈ LEFT ($l_{j(m)}$ ∈ RIGHT) then both $P_{i(m-1),i(m+1)}$ and $P_{j(m-1),j(m+1)}$ are LR-decreasing (LR-increasing) problems. Let EI = { $l_{i(m)}$ ¦ 1 ≤ m ≤ k } and EJ = { $l_{j(m)}$ ¦ 1 ≤ m ≤ k }. We partition EI into EI_n and EI_c, and EJ into EJ_n and EJ_c as follows:

EI_n = { $l_{i(m)}$ ¦ LEN($l_{i(m)} \cup l_{j(m)}$) < X }; EI_c = { $l_{i(m)}$ ¦ LEN($l_{i(m)} \cup l_{j(m)}$) ≥ X };

$$EJ_n = \{ l_{j(m)} \mid \mathrm{LEN}(l_{i(m)} \cup l_{j(m)}) < \mathrm{X} \}; \text{ and } EJ_c = \{ l_{j(m)} \mid \mathrm{LEN}(l_{i(m)} \cup l_{j(m)}) \geq \mathrm{X} \}.$$

Obviously, $l_{i(m)} \in EI_n$ ($l_{i(m)} \in EI_c$) iff $l_{j(m)} \in EJ_n$ ($l_{j(m)} \in EJ_c$). If $l_{i(m)} \in EI_c$ then we say the *match for* $l_{i(m)}$ is $l_{j(m)}$. Similarly, if $l_{j(m)} \in EI_c$, the match for $l_{j(m)}$ is $l_{i(m)}$. If l is the match for l' and l' is the match for l, then we say that *l and l' form a matching pair.*

The following procedure finds a match for each of the elements in set $EI_n \cup EJ_n$.

```
procedure MATCH
  ES ← $\emptyset$;
  for m ← 1 to k do
    if $l_{i(m)} \in EI_n$ and $P_{i(m-1),i(m+1)}$ is a LR-decreasing (LR-increasing) problem then
      find a line segment $l \in$ LEFT (RIGHT) such that $l \notin$ ES ∪ EI ∪ EJ, $y_{i(m-1)} < y(l) \leq y_{j(m)}$,
        and $R(l) \geq x(\mathrm{lm}(P_{i(m-1),j(m)}))$ ( $L(l) \leq x(\mathrm{rm}(P_{i(m-1),j(m)}))$ );
    Let $l_{j'(m)}$ = HLS(y($l$),L(S),x(lm($P_{i(m-1),j(m)}$))) ( HLS(y($l$),x(rm($P_{i(m-1),j(m)}$)),R(S)) )
    ES ← ES ∪ { $l_{j'(m)}$ };
    Let $l_{i(m)}$ and $l_{j'(m)}$ be a matching pair;
  endfor

  for m ← 1 to k do
    if $l_{j(m)} \in EJ_n$ and $P_{j(m-1),j(m+1)}$ is an LR-decreasing (LR-increasing) problem then
      find a line segment $l \in$ RIGHT (LEFT) such that $l \notin$ ES ∪ EI ∪ EJ, $y_{i(m)} \leq y(l) < y_{j(m+1)}$,
        and $L(l) \leq x(\mathrm{rm}(P_{i(m),j(m+1)}))$ ( $R(l) \geq x(\mathrm{lm}(P_{i(m),j(m+1)}))$ );
    Let $l_{i'(m)}$ = HLS(y($l$),x(rm($P_{i(m),j(m+1)}$)),R(S)) ( HLS(y($l$),L(S),x(lm($P_{i(m),j(m+1)}$))) )
    ES ← ES ∪ { $l_{i'(m)}$ };
    Let $l_{i'(m)}$ and $l_{j(m)}$ be a matching pair;
endfor
end of procedure MATCH
```

Lemma 5 and lemma 6 can be used to prove that ES can be constructed, i.e., the line segment with the desired properties can always be found. Let T = EI ∪ EJ ∪ ES. Let p = $|EI_n|$ (note that $|EJ_n| = |EI_n|$). Since |EI| = |EJ| = k and |ES| = 2 * $|EI_n|$, we know that |T| = 2*k + 2*p. It is simple to verify that | T ∩ LEFT| = |T ∩ RIGHT| = k + p and T contains k + p matching pairs. Since the length of each matching pair is at least X, we know that LEN(T) $\geq$ (k+p) X.

In phase one of procedure HVH_CUT we introduce a set of horizontal full cuts to partition P into k+p+1 separable subinstances. Remember that set H(I) contains the set of horizontal full cuts introduced in phase one, $H_1(\mathrm{I}) = \mathrm{H}(\mathrm{I}) \cap E_h(\mathrm{I})$, and $H_1'(\mathrm{I}) = \mathrm{H}(\mathrm{I}) - H_1(\mathrm{I})$. We will show that LEN($H_1'(\mathrm{I})$) $\leq$ (k+p) X / 2. In phase three of procedure HVH_CUT we identify a set of guillotine cuts $H_3(\mathrm{I})$. Remember that $H_3(\mathrm{I}) \subseteq E_h(\mathrm{I})$. We will prove that LEN($H_1(\mathrm{I}) \cup H_3(\mathrm{I})$) $\geq$ (k+p) X. Therefore, LEN($H_1'(\mathrm{I})$) $\leq$ 0.5 * LEN($H_1(\mathrm{I}) \cup H_3(\mathrm{I})$) and the 0.5 bound is satisfied.

Let us now order the line segments in T. Note that it is possible that two line segments l and l' of T have the same y-coordinate. However when this happens, the corresponding lines cannot overlap since we are assuming that there are no guillotine cuts. Also, one of the segments is y($l_{i()}$) or y($l_{i'()}$) and the other is y($l_{j()}$) or y($l_{j'()}$). When we compare two elements with the same y-coordinate value, $l_{j(m)}$ or $l_{j'(m)}$ is considered smaller than $l_{i(m)}$ or $l_{i'(m)}$. We sort all line segments in T by their y-coordinates and form the sequence $l_{q(1)}, l_{q(2)}, \ldots, l_{q(2*k+2*p)}$ such that $y(l_{q(1)}) \leq y(l_{q(2)}) \leq \ldots \leq y(l_{q(2*k+2*p)})$. Let $l_{q(0)}$ and $l_{q(2*k+2*p+1)}$ be the bottom and top side of S, respectively.

Lemma 7: $P_{q(w),q(w+2)}$ is separable for $0 \leq w \leq 2*k + 2*p - 1$.
Proof: For brevity the proof is omitted. The proof appears in [6].

□

Let $T_{odd} = \{ l_{q(w)} \mid l_{q(w)} \in \mathrm{T}$ and w is odd $\}$ and $T_{even} = \{ l_{q(w)} \mid l_{q(w)} \in \mathrm{T}$ and w is even $\}$.

For each line segment $l_{q(w)}$ in T we define its *complement* as the line segment $l^c_{q(w)}$, as follows:

if l(q(w)) $\in$ LEFT (RIGHT) **then**
its *complement* is the line segment $l^c_{q(w)}$ = HLS(y($l_{q(w)}$),R($l_{q(w)}$),R(S)) (HLS(y($l_{q(w)}$),L(S),L($l_{q(w)}$))).

Note that the complement of some line segment in T may overlap with another line segment in T. This can happen only when two line segments in T have the same y-coordinate value. Sets T^c_{odd} and T^c_{even} are defined to have the complements of the elements in sets T_{odd} and T_{even}, respectively.

Given any nonempty non-separable problem P without guillotine cuts, we use the following procedure to partition it.

```
procedure HVH_CUT
  (1) Use MARK and MATCH to construct T;
      Order the line segments in T following the rules mentioned above;
      Partition T into $T_{odd}$ and $T_{even}$;
      if LEN($T^c_{odd}$) $\leq$ LEN($T^c_{even}$) then introduce all the line segments in $T^c_{odd}$
                                        else introduce all the line segments in $T^c_{even}$;
  (2) /* for each resulting partition after step (1), introduce a vertical through cut as follows: */
    if LEN($T^c_{odd}$) $\leq$ LEN($T^c_{even}$) then let g = 1 else let g = 0;
    for w = g to 2(k+p) by 2 do
    case
      : $l_{q(w+1)}$ is $l_{i(m)}$ or $l_{j(m)}$ for some m :
        if $l_{q(u+1)} \in$ LEFT then introduce the leftmost vertical through cut in $P_{q(w).q(w+2)}$
                           else introduce the rightmost vertical through cut in $P_{q(w).q(w+2)}$;
      : $l_{q(w+1)}$ is $l_{i'(m)}$ or $l_{j'(m)}$ for some m :
        case
          /* later on we show the existence of the through cuts we introduce at this step */
          : $l_{q(w+1)}$ is $l_{j'(m)}$ for some m and $y_{i(m-1)} < y(l_{q(w+1)}) \leq y_{j(m)}$ :
            if $l_{q(w+1)} \in$ LEFT then introduce a vertical through cut with x-coordinate value
                                   equal to x(lm($P_{i(m-1).i(m)}$)) in $P_{q(w).q(w+2)}$
                               else introduce a vertical through cut with x-coordinate value
                                   equal to x(rm($P_{i(m-1).i(m)}$)) in $P_{q(w).q(w+2)}$;
          : $l_{q(w+1)}$ is $l_{i'(m)}$ for some m and $y_{i(m)} \leq y(l_{q(w+1)}) < y_{j(m+1)}$ :
            if $l_{q(w+1)} \in$ LEFT then introduce a vertical through cut with x-coordinate value
                                   equal to x(lm($P_{j(m).j(m+1)}$)) in $P_{q(w).q(w+2)}$
                               else introduce a vertical through cut with x-coordinate value
                                   equal to x(rm($P_{j(m).j(m+1)}$)) in $P_{q(w).q(w+2)}$.
        endcase
      endcase
    endfor

  (3) for each resulting partitions of step (2), introduce a horizontal guillotine cut if possible.
end of procedure HVH_CUT
```

Let us now establish some important bounds for LEN(H_1(I) $\cup$ H_3(I)) and LEN(H_1'(I)). This bounds are needed to prove Lemma 9. Our result is given by theorem 1.

Lemma 8: Let H_1(I) $\cup$ H_3(I) and H_1'(I) be defined as above. Then,
(i) LEN(H_1'(I)) $\leq$ min{ LEN(T^c_{even}), LEN(T^c_{odd}) } $\leq$ 0.5 * (k+p) * X;
(ii) LEN(H_1(I) $\cup$ H_3(I)) $\geq$ LEN(T_{even}) + LEN(T_{odd}) $\geq$ (k+p) * X; and
(iii) LEN(H_1'(I)) $\leq$ 0.5 * LEN(H_1(I) $\cup$ H_3(I)).

Proof: For brevity the proof is omitted. The proof appears in [6].

□

Lemma 9: $\mathrm{LEN}(E'_h(\mathrm{I})) \leq 0.5 * \mathrm{LEN}(E_h(\mathrm{I}))$ and $\mathrm{LEN}(E'_v(\mathrm{I})) \leq \mathrm{LEN}(E_v(\mathrm{I}))$.
Proof: For brevity the proof is omitted. The proof appears in [6].

□

Theorem 1: $\mathrm{LEN}(E_{ogp}(\mathrm{I})) \leq 1.75\ \mathrm{LEN}(E_{opt}(\mathrm{I}))$.
Proof: For brevity the proof is omitted. The proof appears in [6].

□

DISCUSSION

As pointed out in [2], there is a problem instance I such that $\mathrm{LEN}(E_{ogp}(\mathrm{I})) = 1.5\ \mathrm{LEN}(E_{opt}(\mathrm{I}))$. In this paper we established the bound $\mathrm{LEN}(E_{ogp}(\mathrm{I})) \leq 1.75\ \mathrm{LEN}(E_{opt}(\mathrm{I}))$. We believe that our upper bound cannot be improved by following our proof technique. However, there might be some other way of proving a smaller approximation bound.

REFERENCES

[1] Du, D.Z., and Y.M. Chen, "On Fast Heuristics for Minimum Edge Length Rectangular Partition," Technical Report MSRI 03618-86, Feb. 1986.

[2] Du, D. Z., L. Q. Pan and M. T. Shing, "Minimum Edge Length Guillotine Rectangular Partition," Technical Report, MSRI 02418-86, January 1986.

[3] Gonzalez, T., M.-T. Shing and S.-Q Zheng, "Bounds for Rectangular and Guillotine Partition", Technical Report #TRCS 86-15, CS Dept., UCSB, July 1986.

[4] Gonzalez, T. and S.-Q. Zheng, "Bounds for Partitioning Rectilinear Polygons", Proc. Symp. Computational Geometry, June 1985, pp. 281-287. (also appears as Technical Report #85-22, CS Dept., UCSB, Dec. 1985).

[5] Gonzalez, T. and S.-Q. Zheng, "Approximation Algorithms for Partitioning Rectilinear Polygons", Technical Report # 85-21, CS Dept., UCSB, 1985.

[6] Gonzalez, T. and S.-Q Zheng, "Improved Bounds for Rectangular and Guillotine Partition", Technical Report #TRCS 86-16, CS Dept., UCSB, July 1986.

[7] Lingas, A., "Heuristics for Minimum Edge Length Rectangular Partitions of Rectilinear Figures", 6th GI-Conference, Dortmund, 1983, Lecture Notes in Computer Science, 195 (Spring-Verlag).

[8] Lingas, A., R. Y. Pinter, R. L. Rivest and A. Shamir, "Minimum Edge Length Partitioning of Rectilinear Polygons", Proc. 20th Annual Allerton Conference on Communication, Control and Computing, Monticello, Illinois, Oct. 1982.

[9] Levcopoulos, C., "Minimum Length and Thickest-First Rectangular Partitions of Polygons," Proceedings of the 23rd Allerton Conference on Communication, Control and Computing, U. of Illinois, Oct. 1985.

[10] Levcopoulos, C., "Fast Heuristics for Minimum Length Rectangular Partitions of Polygons," Proceedings of the 2nd Computational Geometry Conference, June 1986.

[11] Rivest, R. L. "The 'PI' (Placement and Interconnect) System", Proc. 19th Design Automation Conference, June 1982.

[12] Shing, M.T., private communication.

Properties and applications of median and rank-order filters: an intuitive discussion

by

Neal C. Gallagher, Jr.
School of Electrical Engineering
Purdue University
West LaFayette, IN 47907
(317) 494-3518

and

Cecilia A. Weber
Gallagher Associates, Inc.
2639 S. River Rd.
West LaFayette, IN 47906
(317) 743-3245

Abstract

Median and other non-linear operators compose a powerful class of operators with many diverse applications in signal processing. Their versatility and superior performance are demonstrated here in a number of unique examples. A VLSI architecture and an optical filtering system, both based on the powerful threshold decomposition property of these operators, are discussed. Also included is a review of definition and properties of the median filter which is the basis of this class of filters.

I Introduction

Operators based on median filters have evolved into the most important single class of nonlinear operators for signal processing applications. The study of median filters dates back to 1971 when John Tukey [1,2] suggested their application in time series analysis. Since that time a steady increase in the understanding and application of these nonlinear filters has lead to the development of more generalized filters such as rank order filters, stack filters and others. These filter operations derive their utility from the fact that their use significantly improves system performance in many applications. Their use has been enhanced by the organization of their properties into a coherent body of theory and by the derivation of a simple hardware architecture for their fast implementation.

Although we are very familiar with problems involving signal independent additive noise where the signal and noise are spectrally dissimilar and therefore yield to linear solutions, there are many cases where the noise is signal dependent or the problem is otherwise not suited to a linear solution. The effective solution of these non-linear problems can be hampered by attempting to linearize the problem or by using linear tools such as spectral analysis. Even a known optimum linear solution can be extremely sensitive to variations in initial assumptions making it impractical in actual use. In contrast, median-like non-linear operators tend to be robust in regard to assumed characteristics of signal and noise. In order to be able to take full advantage of the improved performance of these non-linear operators we must become as familiar with their properties as we already are with the properties of linear operators. In this paper we will examine a number of unique applications of non-linear operators with the hope of spawning an intuition about these filters which can only be gained by experience, and which will further enhance their practical use. We begin with an explanation of terminology.

II The Median Filter

The median filter is one of the simplest, but yet very powerful members of this nonlinear class of operations. To perform median filtering, we slide a window across sampled data. Typically, we choose this window to contain an odd number of sample values. These value are ranked and the middle, or median, of the ranked values is then the filter output as the window slides over the signal. There are a number of interesting comparisons between taking the median of the samples in our sliding window and taking the average the samples in our sliding window.

1. For the median filter, certain signals are unchanged by the operator. We call these invariant signals roots of the filter. These invariant signals have the property that they are locally monotonic. No matter what portion of the signal appears inside the sliding window, the portion inside the window must be either nondecreasing or nonincreasing. Consequently, the degree of smoothness for these root signals increases with window size. We find that roots for large window sizes will also be roots for smaller windows, but roots for small windows will not necessarily be roots for large windows [3,4].

2. If we take the output of a median filter and filter it again and again, using the same window size, we eventually reach a root signal which doesn't change with repeated filterings. Consequently, repeated median filtering causes a signal to converge to a root [4].

   Neither of these two properties are true for simple window linear averaging.

3. The median operator produces a signal which tends to be more statistically well-behaved than does averaging. As an example, suppose the signal samples within a size 3 window are i.i.d. with a Cauchy distribution. As is well known, the average of the values inside the window has exactly the same Cauchy distribution as each of the three original samples. Consequently, the mean value of this average still does not exist as is always true for Cauchy variables. However, if we define a new random variable to be the median of these three Cauchy variables. The mean of this new variable does exist.

In addition to changing the window size there are some interesting properties which can be seen on changing the relationship between the window size and sampling rate for the filter input. We present an example based on an actual system design.

III Phase Detection

Consider a pulse as shown in Fig. 1. The pulse is a pure sinusoid modulated by an envelope. We want to estimate the zero crossing locations of this sinusoid. We will not present detailed information about the entire system, but rather state some constraints. The Signal to noise ratio is very poor. Because of this, the antenna circuit must be very narrow band around the frequency of the sinusoid ω_c. Otherwise the noise would overwhelm the detector. In addition to other types of external noise, impulse noise is a severe problem. An impulse will hit the front end antenna causing it to ring at the center frequency of antenna circuit. The consequences will totally destroy the performance of the phase detector. The problem is to somehow remove the effects of this impulse noise to allow proper phase detection while minimizing the modification of existing circuitry. What we describe here is the solution that was found to this problem, which also represents a good example of the application of median filtering in signal processing.

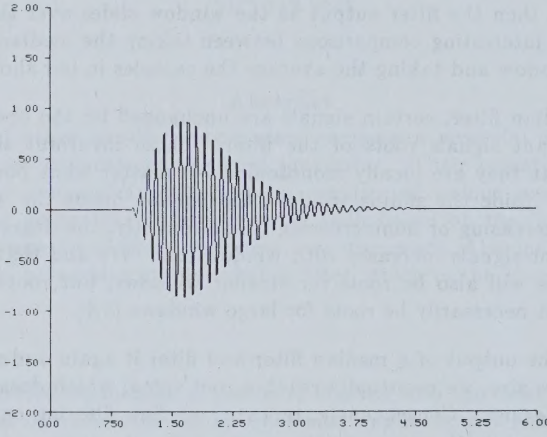


Fig. 1. Noise Free Pulse.

Fitch et. al. [5] have shown that for an analog median filter, zero crossing location is preserved by median filtering. Also, in this same reference it is shown that for an analog filter the sinusoidal response can be identically zero depending on the relationship between the window width and the frequency of the input sinewave. Also it is shown that the response of the discrete median filter approaches that of the analog filter as the sampling rate increases. Consequently, it is determined that the zero crossing information for the desired sinewave could be preserved after median filtering with a size three window, if the sinusoid where sampled at six times per period. The signal is sampled after the narrow band antenna circuit. A problem occurs if the antenna circuit remains narrow band centered around ω_c. Every time an impulse hits the antenna the antenna circuit rings about a center frequency of ω_c as shown in Fig. 2.

So, the center frequency of the antenna circuit is changed to be $2\omega_c$ and the bandwidth is increased to still just include ω_c at the low end of its bandwidth. Now, when an impulse hits the antenna it rings at $2\omega_c$.

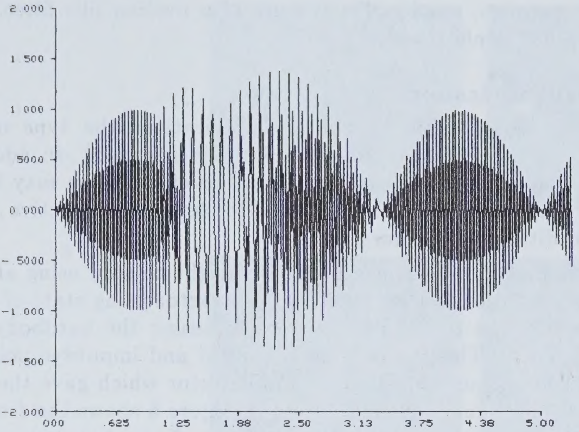


Fig. 2. Antenna Output Due to Signal Plus Impulsive Noise.

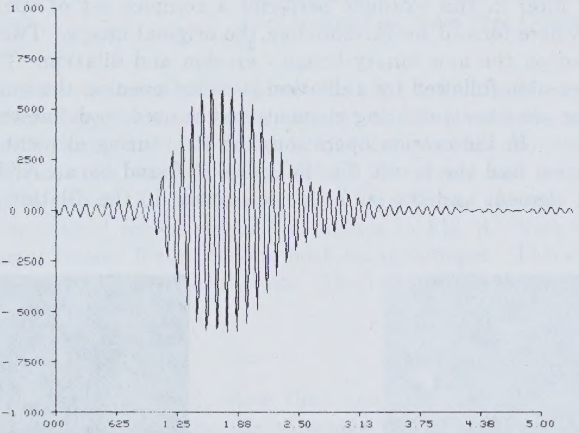


Fig. 3. Filtered Output.

We sample this ringing signal at a rate of $6\omega_c$ and median filter it. This signal is now sampled at a rate of 3 samples per ringing cycle. The median filtered version of this sampled signal greatly reduces the ringing; a continuous median filter could blank out the ringing signal entirely. Then the signal is put through a narrow band filter centered at ω_c. The result of this procedure is that we have greatly reduced the effects of the impulsive noise while maintaining the integrity of the desired signal. See Fig. 3.

This example is interesting because the median filter is used to remove a sinusoidal signal as well isolate the effects of two linear circuits. Not only is this the first example we have seen of where this property of median filters has been used, it is in the design of a commercial consumer system. This example was provided by Patricia Coyle of Information Sciences and Systems, Inc.

In the next section, we describe the use of a median like filter, a stack filter for a timely military application.

IV Hardbody Identification

For orbiting sensors used in military applications the type of noise to be expected in a battle environment may be entirely unknown. In addition to noise from the usual sources, noise caused by high levels of radiation may be very severe. Given this, it may be particularly difficult to design an effective noise filtering scheme using traditional techniques.

In Fig. 4 we find the image of a plume as might be seen using an infrared sensor. The image was provided by a defense contractor using state-of-the-art plume models. Figure 5 shows the original noise free image; the hardbody is artificially inserted for reference. This image is then blurred and impulsive noise is added to simulate the effects of radiation striking the detector which gave the image in Fig. 4. First, we try some simple minded filtering. Figure 6 shows the result of trying a moving average. The window size is 5 x 5. In Fig. 7 we show the result of using a 5 x 5 moving median window. Even in this simple case the median operator is clearly superior to the averaging filter. Now, we show the result of using a stack filter [6]; others may call this a morphological filter [7].

The stack filter in this example performs a complex set of operations. First, binary images where formed by thresholding the original image. Two sets of operations where used on the new binary image - erosion and dilation. The sequence of performing an erosion followed by a dilation is called opening the image. Both erosion and dilation used a structuring element that is used to define the shape of the object of interest. In the erosion operation, the structuring element is centered at each pixel location and the result for that pixel is found by an AND operation of the structuring element and the subimage it covers. In the dilation operation, the

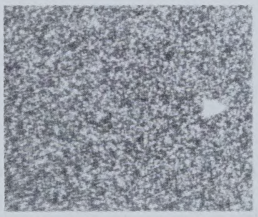


Fig. 4. Distorted, Noisy Plume.

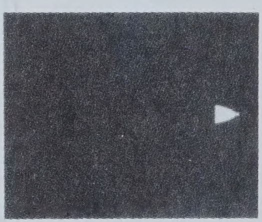


Fig. 5. Original Noise Free Plume with Hardbody Inserted.

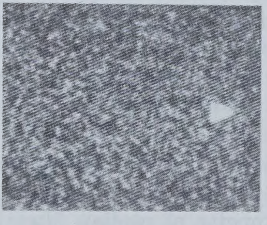


Fig. 6. Moving Average with 5x5 Window.

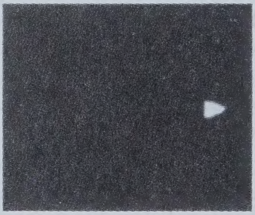


Fig. 7. Moving Median with 5x5 Window.

$$\begin{matrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{matrix}$$

Fig. 8. Binary Structure Element Used in Stack Filter.

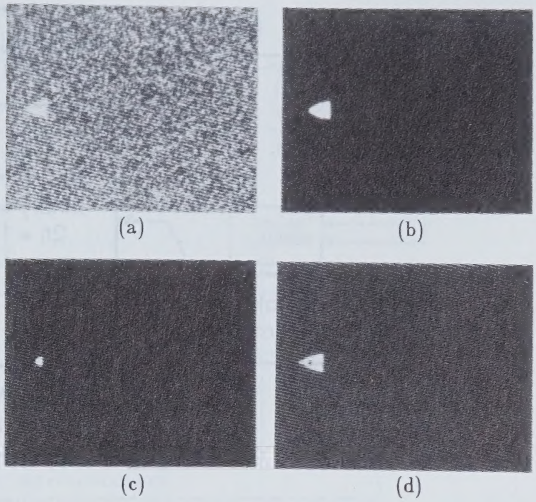


Fig. 9. Stack Filter Output (a) Noisy Input, (b) Entire Plume located, (c) Plume Core Identified, (d) Plume Core imposed on Noise Free Original.

structuring element is centered at the pixels that are 1 (in the binary image). The image region covered by the structuring element is ORed with the structuring element. Two binary images were formed in our experiments from each image by using two thresholds: one was picked to detect the plume and another, a higher one, was picked to detect the plume core. The structuring element used is 9 by 5 with a triangular shaped region set to 1 as shown in Fig. 8. Note that the same structuring element is used for processing both binary images. This example is provide by Henry Chu and Ed Delp of Purdue. The image with the plume core clearly defined is shown in Fig. 9.

V Threshold Decomposition

It is our (admittedly biased) view that among recent developments in the study of these nonlinear operators, the threshold decomposition may be the most significant [8,9]. The threshold decomposition does for median-like nonlinear operators what superposition does for linear operators. It provides a way to decompose a signal into a sum of binary signals; then each binary signal is median (or other) filtered. The resulting filtered binary signals are then added to give the final result. This result is exactly the same as would have been obtained by filtering the original signal without the threshold operation. Figure 10 contains a block diagram of the threshold architecture.

The practical and theoretical implications of the threshold decomposition are significant. Having originally developed this decomposition for median filters, it was found to apply to a large class of nonlinear operations now called stack filters. As it turns out, the class of stack filters is the related to the class of so called morphological filters. The threshold decomposition provides a fast, simple, and flexible VLSI architecture for the implementation of stack (morphological) operators.

To illustrate the power of the threshold decomposition we describe the use of this method in an optical filtering system. This system was designed and built at

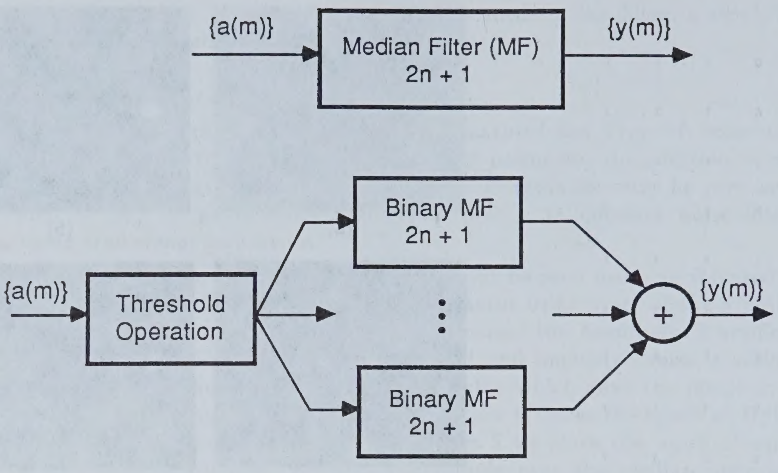


Fig. 10. Threshold Architecture.

Sandia National Laboratory in Livermore, CA, by Ellen Ochoa, Jan Allebach (Purdue), and Donald Sweeney. Details of the architecture have been submitted for publication to Applied Optics.

Threshold decomposition is a method of encoding signals or images in a binary form that allows complicated ranking operations to be replaced by mostly addition operations. The binary signals are derived as the decomposition of the K-level quantized input image into its K-1 component threshold images. The kth threshold image is 1 where the input image exceeds the kth quantizer input threshold and 0 elsewhere. The basic premise is that ranked-order filtering of a multi-level signal is equivalent to decomposing the signal into binary signals, filtering each binary signal, and then adding the individual binary output signals.

The filtering of each binary signal is simply an addition of 1's within the filter window, followed by a thresholding step. The value of the threshold depends on the desired ranked-order operation. The filtering can therefore be implemented as a convolution of the filter window with each threshold component image.

The entire process can be broken down into four steps. Two that are performed electronically and two that are implemented optically. The image is input to the system by use of a video camera. The electronics implements a rising thresholding operation. The thresholded images are multiplexed together onto a Litton iron-garnet light valve. The summing of ones in the filter window is a space-invariant linear operation which is implemented by use of a computer-generated hologram. These summed values are then thresholded again electronically. This step determines which rank-order the filter implements. The final step, which is the addition of these ranked ordered values is again implemented optically. A diagram of this system if shown in Fig. 11.

The utility of the system can be implemented by use of a simple example. Fig. 12 shows an original system input to be median filtered. Fig. 13 shows the output when using the optical system. Fig. 14 shows the output of an all digital processing of the same input image.

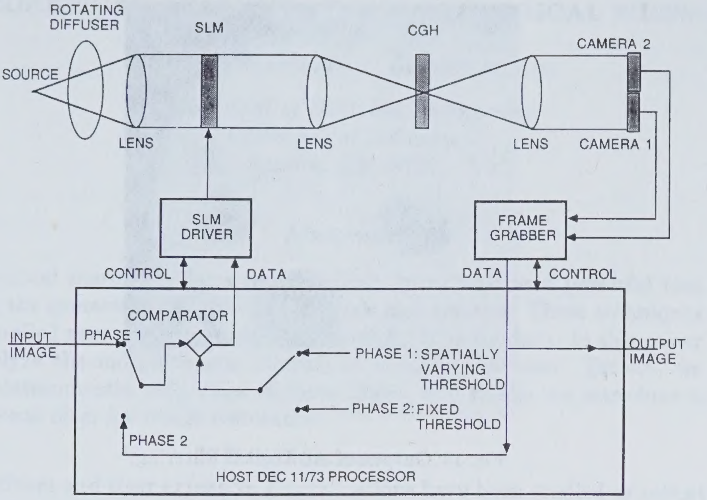

ROTATING DIFFUSER
SOURCE
LENS
SLM
LENS
CGH
LENS
CAMERA 2
CAMERA 1
SLM DRIVER
FRAME GRABBER
CONTROL
DATA
DATA
CONTROL
COMPARATOR
INPUT IMAGE
PHASE 1
OUTPUT IMAGE
PHASE 1: SPATIALLY VARYING THRESHOLD
PHASE 2: FIXED THRESHOLD
PHASE 2
HOST DEC 11/73 PROCESSOR

Fig. 11. Diagram of Optical System.

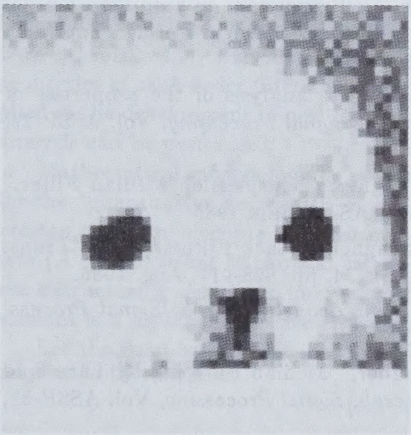


Fig. 12. Original Image to be Median Filtered.

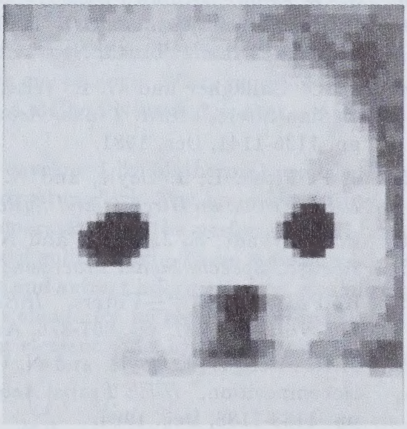


Fig. 13. Output of Hybrid Optical/Digital System.

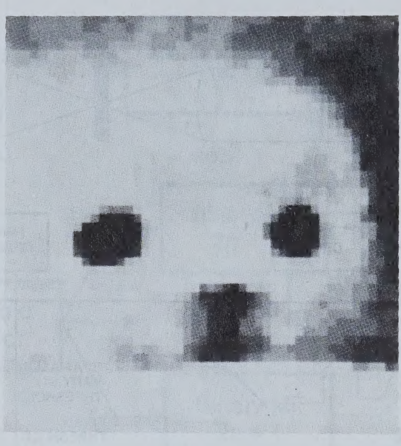


Fig. 14. Output of All Digital Filtering.

References

1. J. W. Tukey, *Exploratory Data Analysis,* Reading, Addison-Wesley, 1971.
2. J. W. Tukey, "Nonlinear (nonsuperposable) methods for smoothing data," in Conf. Rec. 1974 EASCON, pp. 673.
3. S. G. Tyan, "Median Filtering: Deterministic Properties," in *Two-Dimensional Digital Signal Processing II,* T. S. Huang, Editor, Springer-Verlag, Topics in Applied Physics, Vol. 43, New York, 1981.
4. N. C. Gallagher and G. L. Wise, "A theoretical analysis of the properties of median filters," *IEEE Trans. Acoust., Speech, Signal Processing,* Vol. ASSP-29, pp. 1136-1141, Dec. 1981.
5. J. P. Fitch, E. J. Coyle, and N. C. Gallagher, "The Analog Median Filter," *IEEE Trans. on Circuits and Systems,* Vol. CAS-33, Jan. 1986.
6. P. D. Wendt, E. J. Coyle, and N. C. Gallagher, "Stack Filters," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-34, pp. 898-911, Aug. 1986.
7. K. Preston, Jr., "Ξ-Filters," *IEEE Trans. on Acoust., Speech, Signal Processing,* Vol. ASSP-31, pp. 861-876, Aug. 1983.
8. J. P. Fitch, E. J. Coyle, and N. C. Gallagher, "Median Filtering by Threshold Decomposition," *IEEE Trans. Acoust., Speech, Signal Processing,* Vol. ASSP-32, pp. 1183-1188, Dec. 1984.
9. J. P. Fitch, E. J. Coyle, and N. C. Gallagher, "Threshold Decomposition of Multidimensional Ranked Order Operations," *IEEE Trans. on Circuits and Systems,* vol. CAS-32, pp. 445-450, May 1985.

THEORETICAL ANALYSIS OF MORPHOLOGICAL FILTERS*

Robert L. Stevenson and Gonzalo R. Arce

Department of Electrical Engineering
University of Delaware
Newark, DE 19717

Abstract

Mathematical morphology has recently been introduced as a powerful tool for studying the geometrical properties of signals and systems. These techniques have been applied very successfully to the smoothing of noisy data. In this paper we first analyze the morphological filtering of random processes. Second, we derive new deterministic properties of these filters, and finally we introduce a two-dimensional filter for image restoration.

I. Introduction

Median filters and their extensive generalizations have been studied as robust signal smoothers. These non-linear filters have their roots in statistical parameter estimation theory and their analysis is satisfactory from a statistical approach. On the other hand, morphological filters are non-linear filters based on geometrical concepts. These filters can be easily designed to have desirable syntactical performance. The effect of rank-ordered filters on the geometric features of a signal is a topic that is not well understood. Likewise, well known statistical methods cannot be applied to the analysis of morphological filters. The connection between rank-order filters and morphological filters has been established[3], thus there exists a means to better understand the geometrical properties of rank-order filters by studying their morphology. When morphology is applied in the development of new filters however, a method is needed so that a statistical analysis can be performed.

Mathematical morphology is a set algebra developed by Matheron[1] and Serra[2] for the "quantitative description of geometric structures" [2]. The basic operations are *erosion, dilation, opening* and *closing.* These operations can be performed on binary sets ($X \subseteq \mathbf{R}^2$) using Minkowski set addition and subtraction. These transformations use a structuring element to interact with the set and extract information. A structuring element is a set of simpler size and shape and is chosen for its geometric properties.

Let the compact set B denote a structuring element. The symmetric set, $\check{B}$, with respect to the origin is

$$\check{B} = \{-b : b \in B\} = \bigcup_{b \in B} \{-b\}. \tag{1}$$

If $B \subseteq \mathbf{R}^2$ then the symmetric set, $\check{B}$, is obtained by rotating B 180 degrees in the plane. The first operation, erosion, can be viewed geometrically as a set transformation that shrinks a set. It is defined as the set of all points z such that the translation of B by z, B_z, is contained in the original set X. Erosion can be expressed as

$$X \ominus \check{B} = \{z : B_z \subseteq X\} = \bigcap_{b \in B} X_{-b}. \tag{2}$$

The dual with respect to complementation of erosion is dilation.

$$X \oplus \check{B} = (X^c \ominus \check{B})^c. \tag{3}$$

Dilation can be viewed geometrically as a set transformation that expands a set. It can be defined as the set of all points z such that translation of B by z intersects the original set X. This is algebraically the Minkowski addition of the symmetric set $\check{B}$ to X.

$$X \oplus \check{B} = \{z : B_z \cap X \neq \emptyset\} = \bigcup_{b \in B} X_{-b}. \tag{4}$$

Erosion and dilation are in general noninvertible and this fact allows the definition of another pair of dual transformations. If X is eroded by B it is not in general possible to recover X by dilatin g the eroded set by B. This transformation is known as the opening of X with respect to B, it is denoted X_B and defined by

$$X_B = (X \ominus \check{B}) \oplus B. \tag{5}$$

Similarly the dual operation is the closing, denoted by X^B, and it is defined by

$$X^B = (X \oplus \check{B}) \ominus B. \tag{6}$$

Figure 1 shows an example of each operation. The structuring element in this case is the circle shown in (b). Erosion (c) and dilation (d) respectively shrink and expand the image object. The opening (e) transformation filters out the sharp points and edges and the closing (f) fills in the holes and gaps in the object, i.e. they smooth the contours of the set. By combining the opening and closing (g,h) we can obtain smoothed versions of the original image.

To use the morphological operations which are defined for sets, some connections have to be made between sets and functions. To obtain the collection of sets that represent a function, the function $f(z)$ is thresholded at a level t to obtain sets X_t. The collection of sets obtained from thresholding at all possible levels, denoted $\mathcal{X}$, represents the function. The following theorem establishes the connection from sets to functions.

Theorem 1: *(Serra[2]) Let $f(z)$ be a u.s.c. function defined on $\mathbf{R}^n$, and*

$$X_t = \{z \in \mathbf{D} : f(z) \geq t\}, \qquad -\infty < t < \infty. \tag{7}$$

is the set family generated by thresholding $f(z)$ at levels t. Then, the collection of X_i's, denoted by $\mathcal{X}$, are closed and monotonically decreasing, i.e.

$$t' < t \Rightarrow X_{t'} \supset X_t \quad \text{and} \quad X_t = \bigcap_{t'<t} X_{t'} \tag{8}$$

and we have

$$f(z) = \mathrm{Sup}\{t : z \in X_t\}. \tag{9}$$

Conversely, a collection, $\mathcal{X}$, of closed sets generates a u.s.c function $f(z)$ if and only if conditions (8) are satisfied; $f(z)$ is then defined by (9) and satisfies (7).

Once the family of thresholded sets is obtained they can be operated on by a morphological set transformation, $\Psi(.)$, to generate a new family of sets. As long as $\Psi(.)$ is increasing, then the family of transformed sets, $Y_t = \Psi[X_t]$, will satisfy (8) and the multilevel transformed function $y(z) = \psi[f(z)]$ is obtained from (9) $y(z) = \psi[f(z)] = \mathrm{Sup}\{t : z \in Y_t\}$. Thus, a function $f(z)$ can be transformed with the set-processing transformation $\Psi(.)$ by transforming the cross-sections of $f(.)$ and 'piling' the transformed sets; in this way $\Psi(.)$ can be viewed as a function-processing system.

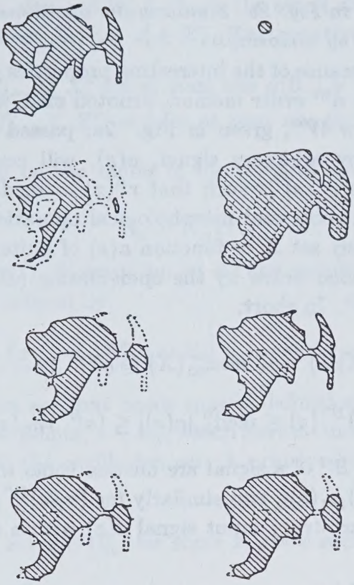


Fig.1 a) Original image object, X; b) Structuring element, B; c) image eroded by B; d) image dilated by B; e) opening of X by B; f) closing of X by B; g) OC of X by B; h) CO of X by B.

In a discrete and quantized signal (image), there are a finite number of points and levels. Hence, the signal representation requires only a finite number of sets. The compact set B will contain a finite number of points, and the inf(sup) taken over the set B will become the min(max) of the elements in the set.

III. One-Dimensional Morphological Filtering

A general morphological filter will consist of several morphological operations performed with various structuring elements $B_i's$. Both syntactical and statistical properties of a filter will depend greatly on the shape and size of the structuring elements as well as the morphological operations performed. Maragos introduced a one-dimensional morphological filter that has some interesting geometric properties in relation to median filters[4]. In particular he proved that a signal is a median root with respect to a window of size $2n+1$, denoted W^n if and only if it is a root of both the opening and closing of a set B^n. Where $B^n \subset W^n$ and $|B^n| = \frac{|W^n|+1}{2}$, and where a root is a signal invariant to further filtering. The sets W^n and B^n are given in Fig. 2 for the one-dimensional filter. It was also proven that an open-closing, or a clos-opening by the same set B^n will give a root with respect to W^n for the median filter. This is appealing since in general it takes several passes of the median to obtain a root signal, and also because the clos-opening (open-closing) is less computationally complex. These observations motivate the following filter definition.

Definition: *The n^{th} order one-dimensional clos-opening filter (1D CO) consists of a closing of the input signal followed by an opening with the structuring element B^n, where B^n is the one-dimensional set show in Fig 2b. Thus the function-processing to be examined will be given by $\psi^n[a(z)] = (a^{B^n})_{B^n}(z)$, The set-processing transform is*

$$\Psi^n(X) = (X^{B^n})_{B^n}, \tag{10}$$

with B^n given in both cases in Fig. 2b. Similarly the one-dimensional open-closing (1D OC) is an opening followed by a closing.□

This filter was chosen because of the interesting properties it shares with the median fiter. Where, ℓ passes of the n^{th} order median, denoted $\mathrm{med}_\ell^n[a(z)]$, is the $n+1^{th}$ rank-order filter over the window W^n, given in Fig. 2a, passed over the signal ℓ times. It has been shown that any arbitrary signal, $a(z)$, will converge to a root. Thus $r^n(z) = \mathrm{med}_\infty^n[a(z)]$, where $r^n(z)$ is such that $r^n(z) = \mathrm{med}_1^n[r^n(z)]$. Two important connection between median roots and morphological operations were made by Maragos[3]; he showed that for any set X or function $a(z)$ of finite extent, the median root with respect to W^n is bounded below by the open-closing (1D OC) and above by the clos-opening (1D CO) by B^n. In short,

$$(X_{B^n})^{B^n} \subseteq \mathrm{med}_\infty^n(X) \subseteq (X^{B^n})_{B^n}, \tag{11}$$
$$(a_{B^n})^{B^n}(z) \leq \mathrm{med}_\infty^n[a(z)] \leq (a^{B^n})_{B^n}(z).$$

The 1D OC and 1D CO by B^n of a signal are median roots with respect to W^n. That is $(a^{B^n})_{B^n}(z) = \mathrm{med}_1^n[(a^{B^n})_{B^n}(z)]$, and similarly for $(a_{B^n})^{B^n}(z)$. Thus a median root signal is obtained from an arbitrary input signal $a(z)$ with a single pass of an 1D OC or of an 1D CO operation.

Deterministic Properties :

We first examine the deterministic properties of the filter. That is, we will determine the structure of the output signal in relation to the input signal structure. The threshold decomposition property is used so that we may examine binary set structures to obtain results for the multilevel signal. If the discrete m-level input signal is $a(z)$, $z \in \mathbf{Z}$, then the $m-1$ thresholded sets can be obtained

$$X_t = \{z : a(z) \geq t\}, \qquad 1 \leq t \leq (m-1). \tag{12}$$

By transforming each set we obtain the collection of output sets, $\mathcal{Y}$ from

$$Y_t = \Psi^n(X_t) = (X_t^{B^n})_{B^n}, \tag{13}$$

where B^n is given in Fig. 2. The multilevel output signal can then be obtained from (9) and is given by $y(z) = \max\{t : z \in Y_t\}$. We first define some important one-dimensional set structures.

The *constant set*, (Ω_n), contains all the points that have $n+1$ adjacent identically valued points. Let $\Omega_n^1 \subseteq \Omega_n$ indicate a positive constant set, that is, all points $z \in X_t$ such that there are n adjacent points, z_i''s, to z which belong to X_t, i.e. $z_i' \in X_t$. Thus the positive constant set is defined by

$$\begin{aligned} \Omega_n^1 &= \{z : B_{z-y}^n \subseteq X_t \text{ for at least one } y; 0 \leq y \leq n+1\}, \\ &= \{z : |B_{z-y}^n \subseteq X_t| = n+1 \text{ for at least one y}; 0 \leq \mathrm{y} \leq \mathrm{n}+1\}. \end{aligned} \tag{14}$$

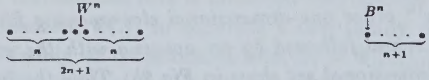


Fig.2 a) Median filter window,W^n; b) structuring element, B^n.

The negative constant set, denoted $\Omega_n^0 \subseteq \Omega_n$, is the set of all $n+1$ adjacent points, z_i''s, that do not belong to the set X_t, i.e. $z_i' \in X_t^c$. The negative constant set is defined by

$$\begin{aligned} \Omega_n^0 &= \{z : B_{z-y}^n \subseteq X_t^c \text{ for at least one } y; 0 \leq y \leq n+1\}, \\ &= \{z : \left|B_{z-y}^n \subseteq X_t^c\right| = 0 \text{ for at least one } y; 0 \leq y \leq n+1\}, \end{aligned} \tag{15}$$

and the constant set will be the union of the positive constant set and the negative constant set,

$$\Omega_n = \Omega_n^1 \cup \Omega_n^0 \tag{16}$$

An *oscillation set*, (Λ_n), is made up of all the points that do not belong to the constant sets. It will be defined by

$$\Lambda_n = \{z : z \notin \Omega_n\} = \{z : 0 < \left|B_{z-y}^n \cap X_t\right| < n+1 \text{ for all } y; 0 \leq y \leq n+1\}. \tag{17}$$

A subset of the oscillation set that needs special definition is the *impulse set*, Υ_n. An impulse set contains those points, $z \in \Lambda_n$, which have no more than $n-1$ adjacent points, z_i''s, which also belong to the oscillation set. A positive impulse, denoted $\Upsilon_n^1 \subseteq \Upsilon_n$, is the set

$$\Upsilon_n^1 = \{z : z - y \in \Omega_n^0, z + x \in \Omega_n^0, \text{ for some } x, y > 0 \text{ such that } y + x = n+1\}, \tag{18}$$

and a negative impulse, Υ_n^0, is

$$\Upsilon_n^0 = \{z : z - y \in \Omega_n^1, z + x \in \Omega_n^1, \text{ for some } x, y > 0 \text{ such than } y + x \leq n+1\}. \tag{19}$$

The impulse set will then be the union of the positive impulse set and the negative, $\Upsilon_n = \Upsilon_n^1 \cup \Upsilon_n^0$. Each point of an arbitrary input set, X_t will thus belong to either the positive constant set, Ω_n^1, or the oscillation set, Λ_n, and those points in the oscillation set can be also in a positive or negative impulse set, Υ_n^1 or Υ_n^0. Furthermore, each point that is not in the input set will belong to either the negative constant set, Ω_n^0, or the oscillation set, Λ_n, and those in the oscillation set can also belong to the negative or positive impulse set. An example of an arbitrary input set is given in Fig. 3, and the corresponding subsets for a second order filter are shown.

Notice that any arbitrary input signal can be split into the five independent sets: Ω_n^1, Ω_n^0, $\Lambda_n \cap \Upsilon_n^c$, Υ_n^1, and Υ_n^0.

We next need to determine how each particular set structure will be affected by the filter transformation. These results are summarized in Table I. The results for the OC operation are also included. Notice that the only difference between the 1D CO and the 1D OC transform is the effect on the set $\Lambda_n \cap \Upsilon_n^c$. The set $\Lambda_n \cap \Upsilon_n^c$ will also be the only set for which the output of the median, 1D OC and 1D CO filters differ. For the single pass median the elements of set $\Lambda_n \cap \Upsilon_n^c$ will become members of any one of the defined sets and for the root signal, members of $\Lambda_n \cap \Upsilon_n^c$ will become members of either Ω_n^0 or Ω_n^1.

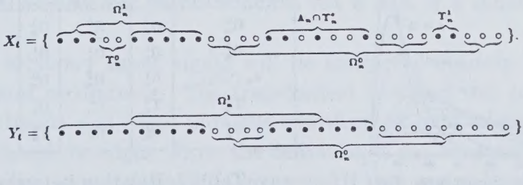


Fig.3 Input set, X^t, and output set, Y^t; for $n = 2$.

Output Statistics

To more fully understand the filtering process it is desirable to determine the output distribution in terms of the input distribution. This description will give an idea of the noise suppression possible and the biasing effects of the filter. From the deterministic properties of the 1D CO filter (i.e. table I) the output statistics can be found for independent, identically distributed signals (i.i.d.) and is given by the next property.

Property 1 : Let $a(z)$ and $y(z)$ be the multilevel i.i.d. input, and output signals, respectively, of the transformation (10), then

$$\begin{aligned} F_y^n(t) =& (n+1)F(t)^{n+1} - nF(t)^{(n+2)} + F(t)^{2(n+1)}(1-F(t)) \\ &+ \frac{(n+2)(n+1)}{2} F(t)^{2(n+1)}(1-F(t))^2. \end{aligned} \tag{20}$$

where $F(.)$ and $F_y^n(.)$ are the input and output distribution functions of the n^{th} order filter, respectively. ▫

Similar results can be found for OC transformations. For an uniform input density, $f(t)$, the output density functions are plotted in Fig 4, for the 1D CO transformation of orders $n =$ 1 – 5, 10, 15, 20.

IV. Two Dimensional Morphological Filters

Due to the fact that openings and closings preserve positive and negative structures, respectively, we define a filter as a combination of openings and closing with one dimensional lines as its structuring elements. The structuring elements are illustrated in Fig. 5, each set has $n+1$ elements where n is the order of the filter, the origin of each set is defined as the point at which all the sets intersect.

To retain positive line structures the opening is performed with each structuring element on the original image and then a union operation is applied to these to obtain the output image. The operations cannot be performed sequentially (i.e. opening by B_1^n, followed by opening by B_2^n, etc.) because opening by any one of the elements will remove the structures from the other orientations. Thus each opening is performed on the original image and the union of these openings combines the structures preserved in each orientation. The closing is used to retain negative line structures in the image. We will refer to this morphological filter as a 2D CO filter, and the binary transformation will be

$$Y_t = \Psi^n(X) = \bigcup_{B_j^n \in J^n} \left(\bigcap_{B_i^n \in J^n} X^{B_i^n} \right)_{B_j^n}, \tag{21}$$

where J is the collection of sets illustrated in Fig. 5. The multilevel transformation is

$$y(z) = \psi^n[a(z)] = \left\{ \max_{B_j^n \in J^n} \left[\min_{B_i^n \in J^n} (a^{B_i^n}) \right]_{B_i^n} \right\}(z). \tag{22}$$

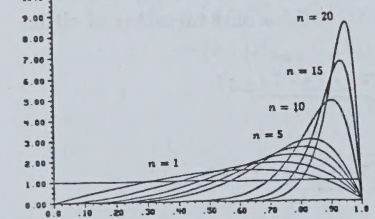


Fig.4 Output density functions for 1D CO transformations, n=1-5,10,15,20.

Input Structures	X^{B^n}	$(X^{B^n})_{B^n}$	X_{B^n}	$(X_{B^n})^{B^n}$	med_1^n	med_∞^n
Ω_n^0	Ω_n^0	Ω_n^0	Ω_n^0	Ω_n^0	Ω_n^0	Ω_n^0
Ω_n^1	Ω_n^1	Ω_n^1	Ω_n^1	Ω_n^1	Ω_n^1	Ω_n^1
$\Lambda_n \cap \Upsilon_n^c$	Ω_n^1	Ω_n^1	Ω_n^0	Ω_n^0	Ω_n, Λ_n	Ω_n
Υ_n^1	Υ_n^0	Ω_n^0	Ω_n^0	Ω_n^0	Ω_n^0	Ω_n^0
Υ_n^0	Ω_n^1	Ω_n^1	Υ_n^0	Ω_n^1	Ω_n^1	Ω_n^1

Table I. Relation between input and output structures.

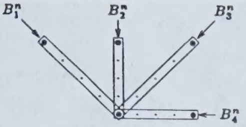


Fig.5 Structuring elements for the two dimensional filter.

Deterministic Properties

Since we are using one dimensional structuring elements, the properties will be very similar to that of the 1D CO filter. We let $a(z)$, $z \in Z^2$ be the discrete m-level input signal, and the thresholded sets can be obtained from (12), the binary transformation is given by (21). As we did with the 1D CO filter we will begin with defining some important set structures.

For the 2D CO filter the constant set (Ω_n) contains all the points that have $n+1$ adjacent identically valued points along any particular direction. The positive constant set is defined by

$$\begin{aligned}\Omega_n^1 =&\{z : (B_i^n)_{z-y} \subseteq X_t \text{ for at least one } B_i^n \text{ and one } y; 0 \le y \le n+1\},\\ &= \{z : |(B_i^n)_{z-y} \cap X_t| = n+1 \text{ for at least one } B_i^n \text{ and one } y; 0 \le y \le n+1\}.\end{aligned} \tag{23}$$

Similarly the negative constant set is defined by

$$\begin{aligned}\Omega_n^0 =&\{z : (B_i^n)_{z-y} \subseteq X_t^c \text{ for at least one } B_i^n \text{ and one } y; 0 \le y \le n+1\},\\ &= \{z : |(B_i^n)_{z-y} \cap X_t| = 0 \text{ for at least one } B_i^n \text{ and one } y; 0 \le y \le n+1\}.\end{aligned} \tag{24}$$

Once again the oscillation set, (Λ_n), is made up of all the points that do not belong to the constant set. It will be defined by

$$\begin{aligned}\Lambda_n =&\{z : z \notin \Omega_n\},\\ &= \{z : 0 < |(B_i^n)_{z-y} \cap X_t| < n+1 \text{ for all } B_i^n \text{ and } y; 0 \le y \le n+1\}.\end{aligned} \tag{25}$$

With a single pass of the 2D CO filter all the points in the oscillation set will become members of the constant set. Although not proven here because of space limitations, we state some properties of this filter. First, if $z \in \Omega_n^1$ then $z \in Y_t$, where Y_t is given in (21). Also, if $z \in \Omega_n^0$ then $z \notin Y_t$. Moreover, if $z \in \Lambda_n$ in the input signal X_t, then $z \in \Omega_n$ in the output signal. Thus we see that after a single pass of the 2D CO filter every oscillation point will become part of the constant set, and thus the transformed original signal will be invariant to further processing. For multilevel signals the important signal structures are defined below.

A *Constant neighborhood* is a region of at least $n+1$ consecutive points in a line, all of which are identically valued.

A *Edge* is a monotonically rising or falling set of points in a line surrounded on both sides by constant neighborhoods

An *Oscillation* is a sequence of points which is not a part of a constant neighborhood, or an edge.

As before an arbitrary input signal will be composed entirely of constant neighborhoods, edges and oscillations. The transformed function will contain the original constant neighborhoods and edges plus the transformed oscillations, which are either constant neighborhoods or edges. From the definition of the constant neighborhood, we know that all thin line structures will be preserved if they are at least $n+1$ points long. Also for an object to be preserved it must have at least in one dimension $n+1$ points.

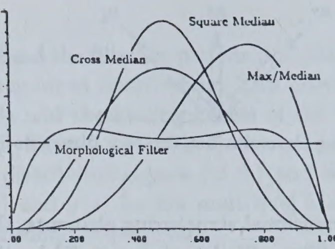


Fig.6 Output density functions for 2D CO transformations n=1-5,10,15,20.

Output Statistics:

Without being able to define how oscillations are transformed into constant sets, the output statistics for the general order filter cannot be found as we did for the one-dimensional filter. Instead of doing such an analysis, in this section we will compute the output statistics for the first order 2D CO filter and compare this with the other two-dimensional filters. Higher order output statistics are found through monte carlo trials.

We can compute the output statistics for the first order filter because it is possible to determine how oscillations will be transformed. For the first order filter a point will be a member of the constant set if it is adjacent to an identically valued point. Thus if $z \in \Lambda$ and $z \in X_t \quad \Rightarrow \quad z \notin Y_t$, and $z \in \Lambda$ and $z \notin X_t \quad \Rightarrow \quad z \in Y_t$. From this information we can determine the output statistics for the first order filter on binary signals. The output distributions for the multilevel follows easily and is given in the next property.

Property 3 : Let $a(z)$ and $y(z)$ be the multilevel i.i.d. input and output signals, respectively, of the transformation (22), then

$$F_y^1(t) = F(t)^8(1 - F(t)) + F(t) - F(t)(1 - F(t))^8, \tag{26}$$

where $F(t)$ is the input distribution function and $F_y^1(t)$ is the output distribution of the first order function.

In Fig. 6 the output distributions for the first order filter is plotted for a uniformly distributed input signal, also plotted are first order output distributions for the square median, cross median[6] and the max/median[9]. From this plot it is easy to see that the square median has the most noise suppression followed by the max/median and the cross median. The 2D CO filter will suppress the least amount of noise for a given order filter. For higher order filters the output distribution of the 2D CO was found through monte carlo trials. These results show that for $n > 1$, the 2D CO is biased.

Performance Comparisons

Figure 7(a,b) shows a noiseless and a noisy aerial photograph. The noise is impulsive, where the probability of an impulse occurring is .2. Figure 7(c,d,e,f) shows the second order square median, the second order cross median, the second order max/median, and finally the second order 2D CO filter. Notice that the square median produces a very blurry image, and while the cross median is better, there is still loss of significant signal structure. The max/median is the best at retaining the thin line structures like the bridges. The 2D CO filter is the best at overall structure preservation but fails to suppress all of the noise. Very similar results are obtained for gaussian noise. In passing we mention that unlike other ranked-order filters, by increasing the order of the 2d CO filter, we are able to obtain more noise suppression without seriously degrading the image signal.

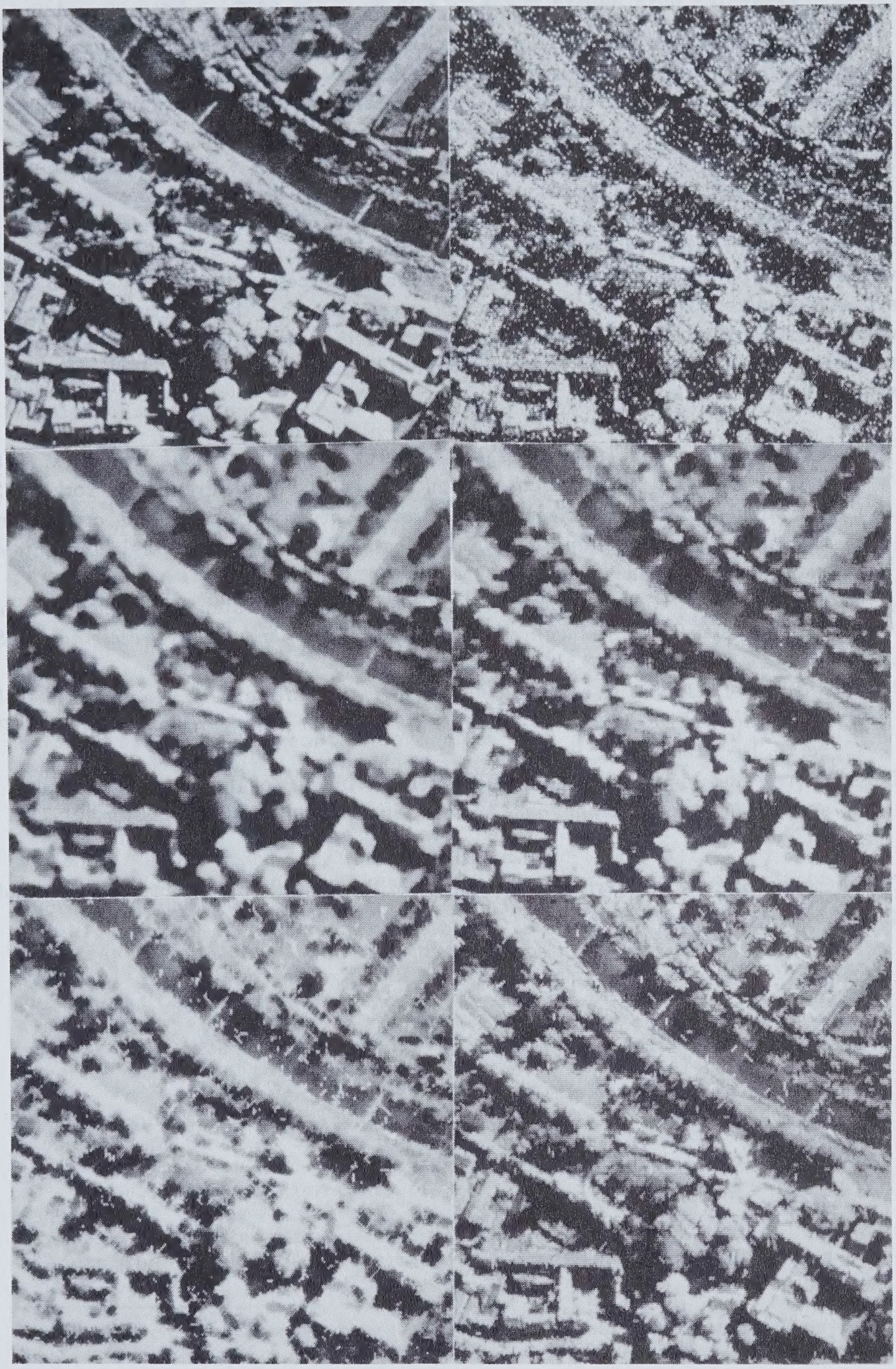


Fig.7 a) Original image; b) Image with impulsive noise; c) square median; d) cross median; e) max/median; f) 2D-CO filter.

V. Conclusion

For one-dimensional filters we introduced a simple morphological filter that would preserve edges in signals. Using morphology we could describe in detail the effect of the filter on the structure of the signal. This analysis also provided a method to calculate the output statistics for the general order filter.

Using morphological operators it was possible to develop a filter which retained both positive and negative line structures in all possible orientations, resulting in a filter that has very good syntactical properties. For equivalent order filters however this new morphological filter, (2D CO), does not have as much noise suppression as any of the median type filters we examined. Thus there is a necessary trade-off between the syntactical and statistical properties. Note however, that since the structure preserving characteristics of the 2D CO filter are much better than any of the other filters, it is possible to use higher order 2D CO filters, to obtain higher noise suppression, without significant structure loss.

References

[1] G. Matheron, *Random Sets and Integral Geometry,* J. Wiley and Sons, New York, 1975.

[2] J. Serra, *Image Analysis and Mathematical Morphology,* Acad. Press, New York, 1982.

[3] P. A. Maragos, *A Unified Theory of Translation-Invariant Systems with applications to Morphological Analysis and Coding of Images,* Ph. D. Thesis, Georgia Institute of Technology, July 1985.

[4] G. R. Arce, N. C. Gallagher, and T. A. Nodes, " Median Filter: Theory for one and two Dimensional Filters," *Advances in Computer Vision and Image Processing,* (T. S. Huang, ed.) Vol. 2, Chap. 3, JAI Press Inc., 1986.

[5] J. P. Fitch, E. J. Coyle, and N. C. Gallagher, Jr., *Median Filtering by Threshold Decomposition. IEEE Trans. on Acoustics, Speech, and Signal Processing,* Vol. ASSP-32, No. 6, Dec. 1984.

[6] G. R. Arce and M. P. McLoughlin, "Theoretical Analysis of the Max/Median Filter," *To appear in the IEEE Trans. on Acoustics, Speech, and Signal Processing,* 1987.

Acknowledgement

The authors gratefully acknowledge the support of the Reston-Sperry Corporate Technology Center.

A CLASS OF RANK FILTERS BASED ON THE WILCOXON SIGNED RANK STATISTIC

IICKHO SONG and SALEEM A. KASSAM
Department of Electrical Engineering
University of Pennsylvania
Philadelphia, PA 19104

ABSTRACT: Starting from the observation that many edge-preserving and impulse-rejecting nonlinear filters can be considered to belong to a general class of filters with a particular structure, a dual class of nonlinear filters are defined. It is shown that a subclass of rank-filters (R-filters), the Wilcoxon filters, can be derived as a special case of this class of nonlinear filters. The Wilcoxon filter which has basis in the Wilcoxon signed rank statistic of statistical theory is next modified to have better performance characteristics for edge-preservation, impulse-rejection and detail-retention in addition to Gaussian noise reduction. Performance of some well-known nonlinear filters, including the improved version of the Wilcoxon filter, are compared through computer simulations.

1. INTRODUCTION

A variety of nonlinear filters including those which have basis in robust estimation techniques of statistical theory have recently been considered and shown to have fairly good performance characteristics in restoring signals with edges from observations corrupted by both Gaussian and impulsive (heavy-tailed) noise. These nonlinear filters have several desirable characteristics such as impulse rejection and edge preservation, which can not generally be obtained by an application of linear filters. Furthermore, since Gaussian noise may most effectively be reduced by a linear (averaging) operation, it is apparent that nonlinear filters which exhibit some linear characteristics will have better overall performance in smoothing signals containing edges with additive Gaussian noise and impulses. Among the nonlinear filters that can include an averaging operation are the order-statistic (OS) filters or the L-filters [1,2], the M-filters [2], the k-nearest neighbor (KNN) filters [3], the symmetric nearest neighbor (SNN) mean filters, the symmetric nearest neighbor (SNN) median filters [4] and the linear median hybrid (LMH) filters [5,6].

It is interesting that many of these nonlinear filters can be considered to belong to one general class of filters. The filter transformation of this general class of nonlinear filters consists of a nonlinear (vector) operation on subsets of observations in a fixed window followed by a linear operation. This class includes, for example, the L-filters and other filters as special cases. For any given filter in this general class the linear and nonlinear operations can be interchanged to produce a dual filter. This second class of dual filters also includes several common filters as special cases. For example, the running mean filter, the median filter, and SNN-median filters are among the special cases of this second class of nonlinear filters. Note that these two classes of filters are not disjoint. Any given filter may be considered to belong more naturally to one of these two classes.

A more interesting property of the second class of filters is that a subclass of filters which have basis in the rank-estimates (R-estimates) of robust estimation techniques can easily be derived as a special case, whereas the L-filters which also have basis in the L-estimates of robust estimation techniques may be naturally categorized into the first class of

---

This research is supported by the Air Force Office of Scientific Research under Grant 82-0022.

nonlinear filters. Noting that L- and M-estimates of statistical theory have recently been applied in signal processing to yield the L- and M-filters, respectively, this formulation of R-filters from a general class of nonlinear filers is quite interesting.

More specifically, the Wilcoxon filter will be derived as a special case of the second class of nonlinear filters. Though the Wilcoxon filter does perform effectively for edge preserving and impulse rejecting smoothing of signals [7], it is possible to modify the Wilcoxon filter to yield a new class of filters, the limited-degree extended-averaging Wilcoxon filters (LEWF), which have better performance characteristics. Furthermore, the LEW filters can generally preserve details more completely than the original Wilcoxon filter. Another interesting property of the LEW filters is that although they are derived as a special case of the second class of nonlinear filters, the LEW filters are closely related to the Winsorized rank estimates of statistical theory [8,9].

Performance comparison results from computer simulations will also be given in Section 4 using several noisy versions of a one-dimensional signal and an image as inputs to several filters.

2. TWO CLASSES OF NONLINEAR FILTERS

Many of the previously considered filters, which may or may not have basis in robust estimates of statistical theory, are non-trivial subclasses of a more general class of filters whose filter tranformation $T_1(X)$ can be expressed as

$$T_1(X) = f_1\{g_1(X)\}, \tag{1}$$

where g_1 is generally a nonlinear vector function mapping N-dimensional space to P-dimensional space, f_1 is generally a non-trivial linear function mapping P-dimensional space to 1-dimensional space and X is the N-dimensional vector of the observations in a window of size N. The P components of $g_1(X)$ may, in many case of interest, be interpreted as each arising from some nonlinear transformation of a subset of the N observations. If g_1 is the median operation and f_1 is the identity function then T_1 represents the median filters; if g_1 is the identity function and f_1 is the average of N values then T_1 represents the running mean filters. By choosing other functions or operations for f_1 and g_1, more interesting filters such as the L-filters, the KNN filters and the SNN-mean filters can be derived from this general class of filters. A particularly interesting nonlinear filter class is obtained with each of the P components of $g_1(X)$ being median$\{X_{m_1}, X_{m_2}, \ldots, X_{m_M}\}$ for $P = {}_N C_M$ possible combinations of subscripts $(m_1, m_2, \ldots, m_M)$ such that $1 \leq m_1 < m_2 \ldots < m_M \leq N$, and with f_1 the averaging function. In this case $T_1(X)$ represents a class of L-filters with nonequal symmetric coefficients [10]. More explicitly, the filter transformation becomes, for M odd,

$$T_1(X) = \binom{N}{M}^{-1} \sum_{k=0}^{N-M} \binom{\frac{M-1}{2}+k}{\frac{M-1}{2}} \binom{N-k-\frac{M+1}{2}}{\frac{M-1}{2}} X_{(\frac{M+1}{2}+k)} \tag{2}$$

where $\binom{n}{m} = {}_nC_m = n!/(n-m)!m!$ and $X_{(i)}$ is the i-th order statistic in X. This filter transformation becomes, for example,

$$T_1(X) = \binom{7}{5}^{-1}\{6X_{(3)}+9X_{(4)}+6X_{(5)}\}, \tag{3}$$

when N=7 and M=5. It can also be shown that the filters with M=2I and M=2I-1 are exactly the same for given window size N [10].

Let us now consider another class of nonlinear filters which can easily be constructed by an interchange of the linear and nonlinear operations in (1); the resulting class of nonlinear filters has

$$T_2(X) = g_2\{f_2(X)\} \tag{4}$$

as the filter transformation. Here f_2 is a vector linear operation mapping N-dimensional space to P-dimensional space, g_2 is a nonlinear function mapping P-dimensional space to 1-dimensional space and X is the vector of observations in a window of size N. For this second class of nonlinear filters we can also find several well-known nonlinear filters as special cases. As interesting examples, with f_2 the averaging operation, g_2 the median and P=1, T_2 represents the running mean filters; if f_2 is the identity function and g_2 is the median operation, T_2 represents the median filters; other interesting filters such as the SNN-median filters and the LMH filters may be generated from this class of nonlinear filters. In particular, the LMH filter whose output is the median of means of disjoint subsets of the observations in a window can naturally be considered as a special case of this class, when the P components of $f_1(X)$ are averages of contiguous observations in disjoint subsets of X and g_2 is the median function.

3. THE LIMITED-DEGREE EXTENDED-AVERAGING WILCOXON FILTER

As has been noted in the Introduction, among the filters in the first class expressed by (1), the L-filters have a basis in the L-estimates of robustness theory. The class of M-filters is similarly based on the M-estimates of statistical theory. Since these two classes of robust estimates have successfully been applied in signal processing, one can expect that the third class of robust estimates, the class of rank estimates (R-estimates), can similarly be exploited in signal processing applications. In this section a sub-class of R-filters is derived as a special case of the second class of nonlinear filters described in Section 2. The basic idea in the definition of an R-filter is that the ranks of the observations are used in forming the output of an R-filter: therefore, what has more effect on the output of an R-filter is the ranks of observavtions, and not the actual values of the observations. Consequently, very large values (e.g. impulses or values from the other side of an edge (if one exists in a window) when most of the value are from one side of the edge) will have considerably less effect on the output than they would have in linear filters. Thus R-filters generally reject impulses and preserve edges more effectively than the linear filters do.

One particular choice of f_2 and g_2 in (4) which yields a subclass of the rank filters (R-filters) is $f(X)=\{(X_i+X_j)/2,\ 1\leq i\leq j\leq N\}$ and g_2 the median operation. This choice gives us the Wilcoxon filter with filter output for any window

$$T_W(X) = \text{median} \{ (X_i+X_j)/2,\ 1\le i\le j\le N \}. \tag{5}$$

The Wilcoxon filter, which has a basis in the Wilcoxon signed rank statistic [11-12], has very recently been investigated [7] and shown to be reasonably effective in edge preserving, impulse rejecting smoothing of signals.

The performance characteristics of the Wilcoxon filter can be improved since (A) only two values are averaged which limits the linear characteristic and (B) all possible combinations of two values are taken into account among which there exist average values of one value from one side of an edge (if one exists in a data window) with another value from the other side of the edge. (A) implies that the Wilcoxon filter may not reduce Gaussian noise sufficiently and (B) implies that the Wilcoxon filter may give poor performance characteristics for edge-bearing signals.

To obtain filters which have better performance characteristics than the Wilcoxon filter, we modify the Wilcoxon filter as follows: first, we limit the maximum difference between time indices used in the averaging operation, and then we replace the pairwise average with a linear combination of M values. This modification will produce the limited-degree extended-averaging Wilcoxon (LEW) filters with the filter output

$$T_{LEW}(X) = \text{median} \{ \sum_{i=1}^{M} a_i X_{m_i},\ 1\le m_1\le m_2 \ldots \le m_M\le N,\ \max|m_i-m_j|<D \}, \tag{6}$$

where the sum of coefficients a_i is 1. It is expected from observations (A) and (B) and the modifications above that the LEW filters have better overall performance characteristics than the original Wilcoxon filter. It is noteworthy that the performance characteristics of the LEW filters can be controlled by a set of parameters (D,M); if more smoothing is needed, large values of D and M can be used; if a strong nonlinear charcteristic is required because an original signal has many edges or narrow pulses, small values of D and M may be used. Some interesting properties of the LEW filter are:

(a) If D=1 or M=1 and $a_i=1/M$, i=1,2,...,M, the LEWF is the median filter.

(b) If D=N, M=2 and $a_i=1/M$, i=1,2,...,M, the LEWF is the Wilcoxon filter.

(c) If D=M=N and $a_i=1/M$, i=1,2,...,M, the LEWF is almost the running mean filter.

(d) The LEWF with $a_i=1/M$, i=1,2,...,M, is a data-dependent L-filter with at most M out of N coefficients having nonzero values. These coefficients can take values only from the set {0, 1/M, 2/M,...,(M-1)/M, 1}

It is also noteworthy that the LEW filter is quite closely related to the Winsorized rank statistics [8,9], even though the LEWF in this paper was derived differently, from the general form of (4).

A primary difference between the LMH filter and the LEW filter lies in the way in which subsets of observations are used in the linear operation in (4); in the LEW filter all the subsets are of the same size and they are not disjoint, whereas they are generally of different size and composed of disjoint subsets of contiguous observations in the LMH filter.

4. PERFORMANCE COMPARISON OF FILTERS THROUGH COMPUTER SIMULATIONS

In this section performances of several nonlinear filters will be compared through computer simulations.

The one-dimensional original signal of Figure 1-(a) has edges and pulses of different widths. Gaussian noise N(0,1) and 5 impulses of heights approximate 10 are added to this original signal to generate the noisy signal (b) which is then used to obtain outputs of various filters. The ouputs are shown in Figures 1-(c) through (*i*). Figure 2 shows a set of computer simulation results for an image which has several edges and small rectangles (which corresponds to narrow pulses in the one-dimensional case). The original image is of size 100x100 and is composed of bright portions of grey-level 170 and a sine wave background with values ranging from 50 to 110. Zero-mean Gaussian noise with variance 1600 and about 150 impulses of height ±250 are added to the original image, and then truncated at 255 and at 0. Output images of several filters are also shown in Figure 2. In Figures 1 and 2 the numbers in parentheses are window sizes and M and D (for LEWF). The DW-MTM filter output is the average of observations within an interval [m-q,m+q] with m being the sample median of the observations in a smaller window and q being a predetermined parameter.

From Figures 1 and 2 we may conclude that the performance of LEWF is generally quite good. Since the DW-MTM filter first selects a median from a smaller window while the LEWF first averages observations, the edge-preserving characteristic of the DW-MTM is generally better than that of the LEWF when the size of the smaller window of the DW-MTM is small enough and q is selected properly. For the same reason, the averaging characteristic of the LEWF is generally somewhat better than that of the DW-MTM. Since the DW-MTM filter requires more information about the original signal and noise statistics to set the value of q appropriately, it performs generally better than the LEWF if the information is given correctly. Therefore, the LEWF requiring less information can be said to be more robust than the DW-MTM filter whose performance depends on the information about the original signal and noise statistics.

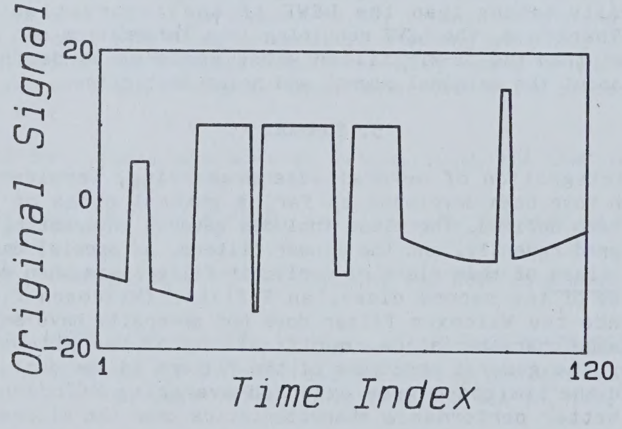
5. SUMMARY

As an integration of several edge-preserving, impulse-rejecting filters which have been developed so far, a general class of nonlinear filters has been defined. The class includes several interesting nonlinear filters designed recently, and the linear filters, as special cases.

A dual class of this class of nonlinear filters was then defined. As a special case of the second class, an R-filter (Wilcoxon filter) was derived. Since the Wilcoxon filter does not generally have sufficiently good performance characteristics, modifications of the Wilcoxon filter motivated by the general structure of the filters in the dual class were made to yield the limited-degree extended-averaging Wilcoxon filters. These have better performance characteristics than the Wilcoxon filter. The introduction of these R-filters complements recent work on nonlinear filters based on the classes of L- and M-estimates of robustness theory.

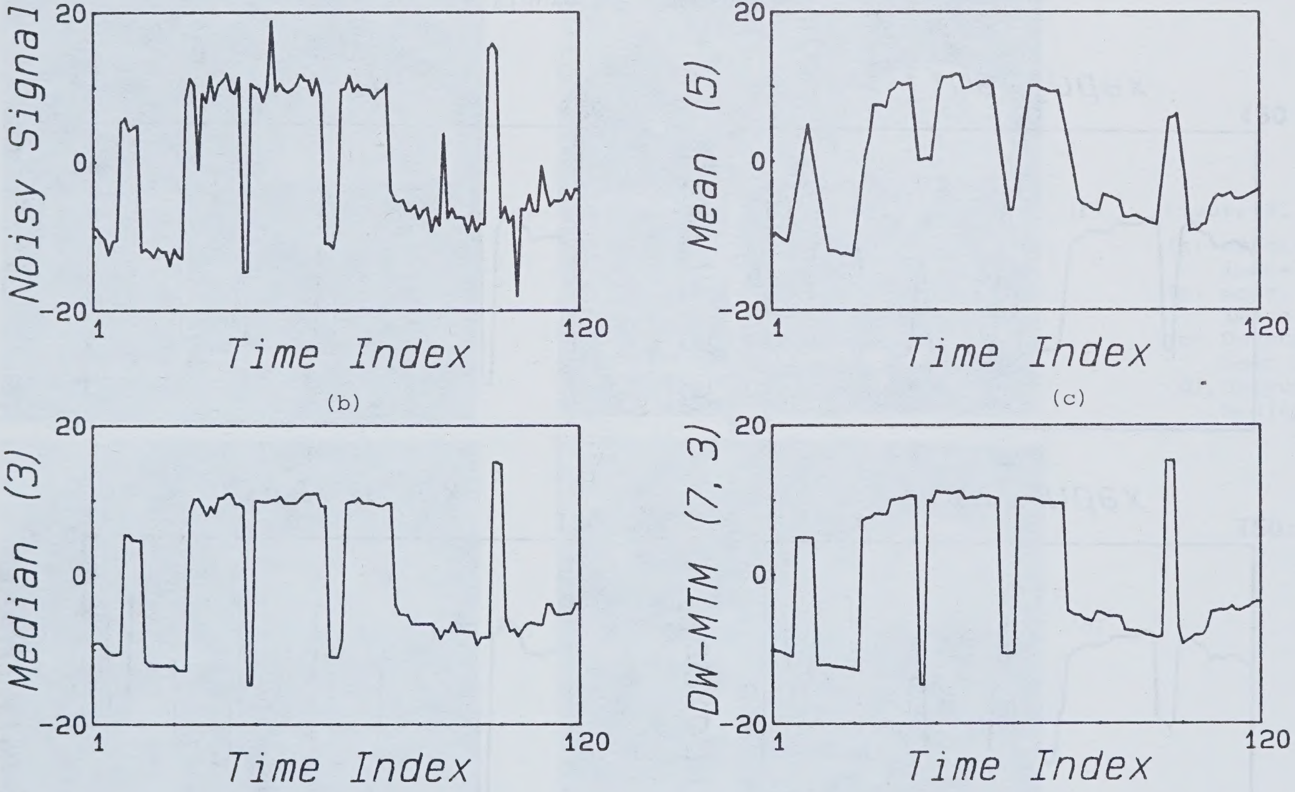
REFERENCES

[1] A.C. Bovic, et al, "A Generalization of Median Filtering Using Linear Combinations of Order Statistics", IEEE Tr. Acous., Speech, Signal Proc., vol. ASSP-31, pp. 1342-1349, Dec. 1983.
[2] Y.H. Lee and S.A. Kassam, "Generalized Median Filtering and Related Nonlinear Filtering Techniques", IEEE Tr. Acous., Speech, Signal Proc., vol. ASSP-33, pp. 672-683, June 1985.
[3] L.S. Davis and A. Rosenfeld, "Noise Cleaning by Iterated Local Averaging", IEEE Tr. Syst., Man, Cyber, vol. SMC-8, pp. 705-710, Sep. 1978.
[4] D. Harwood, et al, "A New Class of Edge-Preserving Smoothing Filters", TR-1397, Computer Vision Lab., Univ. of Maryland, pp. 1-22, Sep. 1978.
[5] P. Heinonen, et al, "Median Type Filters with Linear Substructures", Proc. 1986 ISCAS, pp. 357-360, San Jose, May 1986.
[6] P. Heinonen, Linear Median Hybrid Filters, Doctor of Technology Dissertation, Tampere University of Technology, Finland, Feb. 1986.
[7] R.J. Crinon, "The Wilcoxon Filter : A Robust Filtering Scheme", Proc. ICASSP, pp. 18.5.1-18.5.4, Tampa, March 1985.
[8] P.J. Huber, Robust Statistics, Wiley, New York, 1981.
[9] S.A. Kassam, Statistical Techniques for Signal Processing, Renewal Proposal to the Air Force Office of Scientific Research, July 1986.
[10] I. Song, Nonlinear Techniques for Detection and Filtering of Discrete-time Signals, Ph.D. Dissertation, Dept. of Elect. Engr., Univ. of Pennsylvania, Dec. 1986.
[11] J. Hajek and Z. Sidak, Theory of Rank Tests, Academic, New York, 1967.
[12] T.P. Hettmansperger, Statistical Inference Based on Ranks, John Wiley & Sons, New York, 1984.



(a)

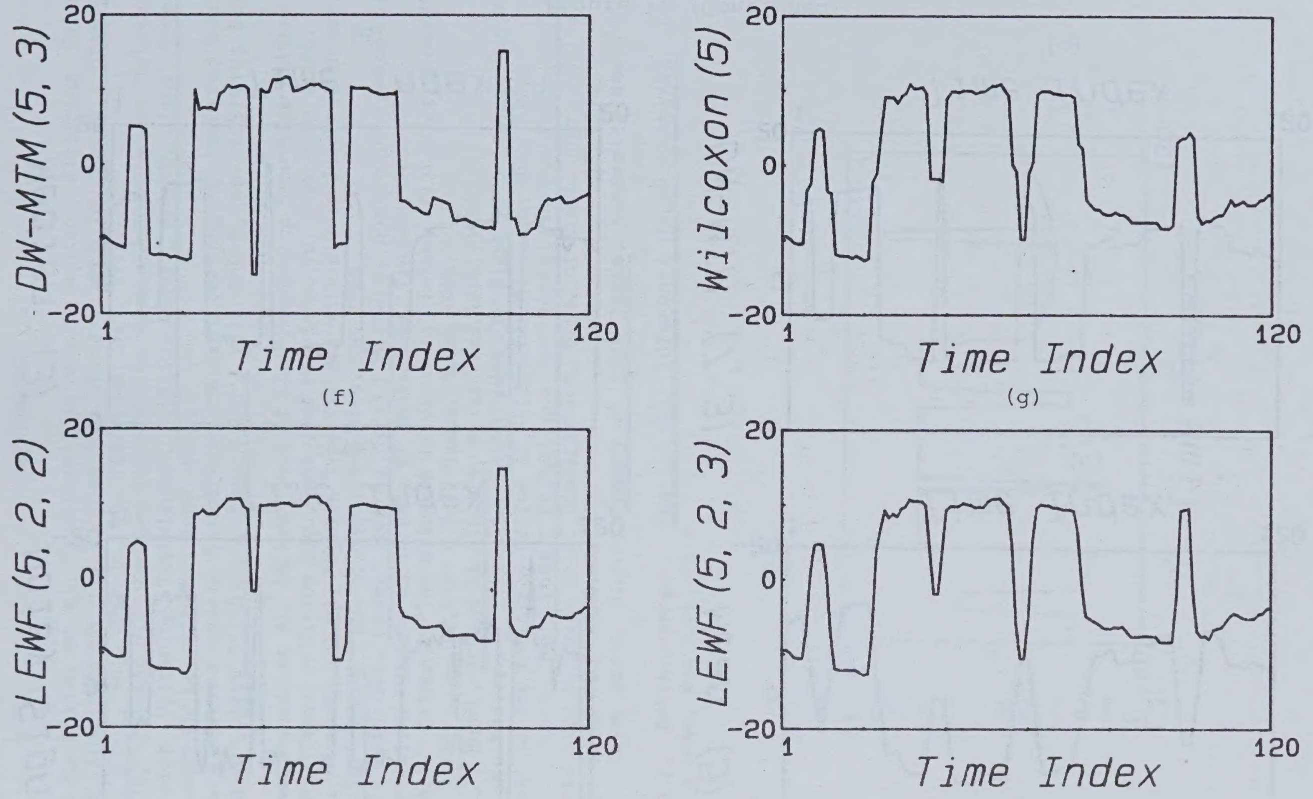
Figure 1. Simulation Results for a One-dimensional Signal.



(b) (c)

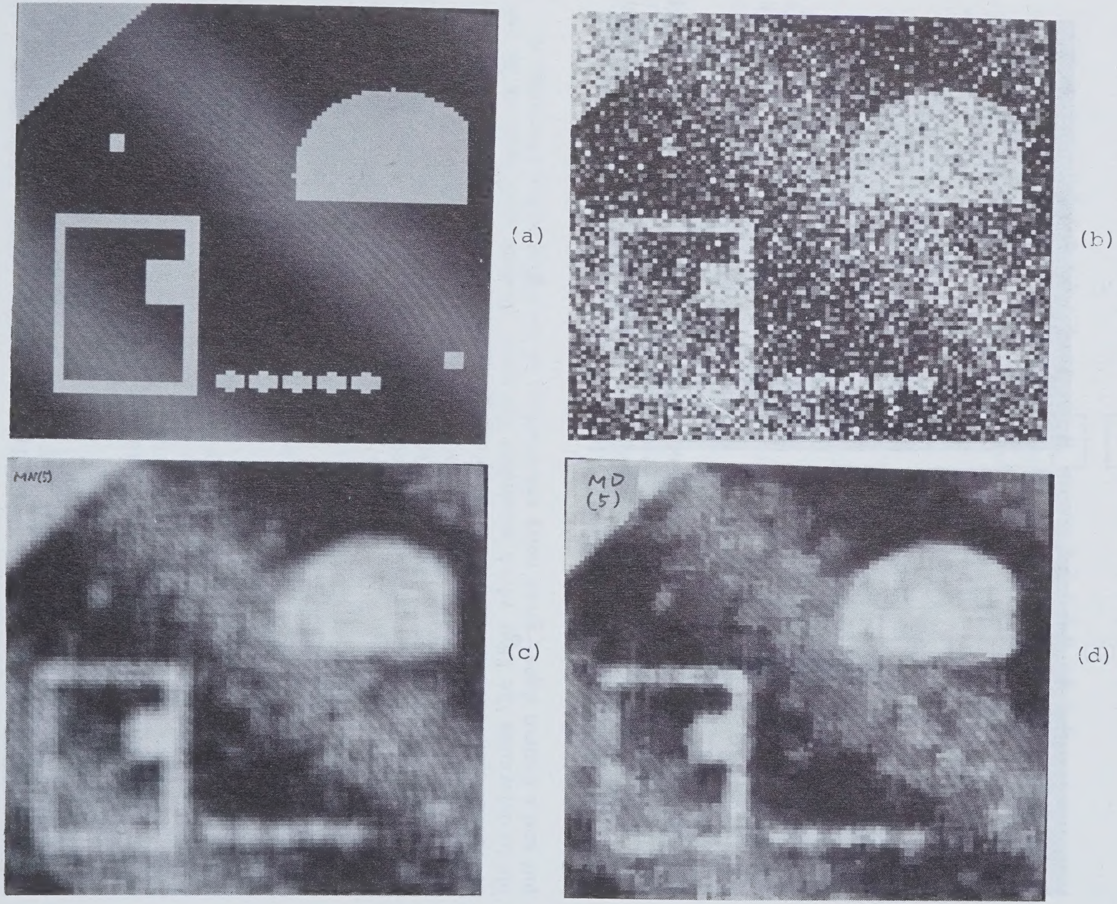
(d) (e)

Figure 1. (Continued)



(f) (g) (h) (i)

Figure 1. (Continued)



(a)

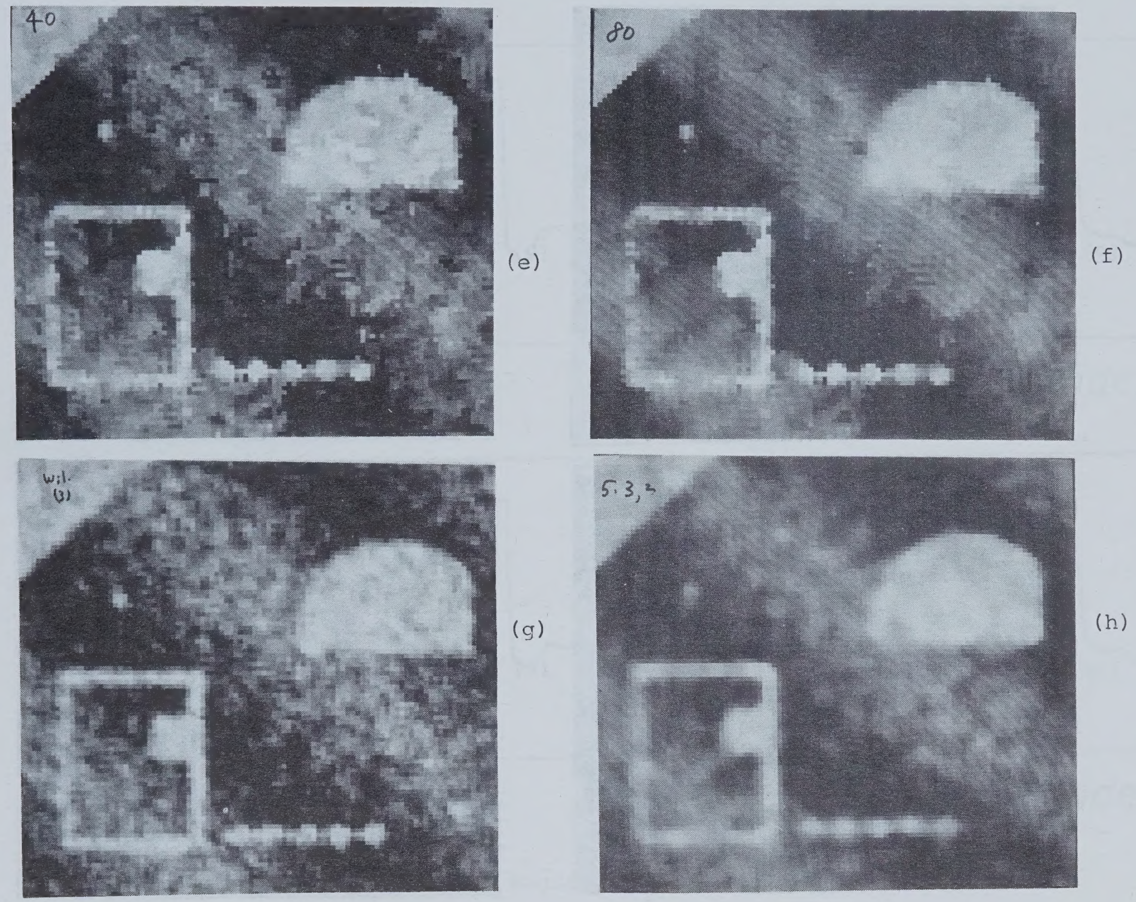
(b)

(c)

(d)

Figure 2.

(a) original image
(b) noisy image
(c) Output of Mean (5)
(d) Output of Median (5)


40
80
wil.
(3)
5.3,2

(e)

(f)

(g)

(h)

Figure 2.
(Continued)

(e) Output of DW-MTM (7,3), q=40
(f) Output of DW-MTM (7,3), q=80
(g) Output of Wilcoxon (3)
(h) Output of LEWF (5,3,2)

CHOOSING THE STACK FILTER WHICH MINIMIZES THE MEAN ABSOLUTE ERROR CRITERION

Edward J. Coyle
School of Electrical Engineering
Purdue University
West Lafayette, IN 47906

Abstract

By definition, the class of stack filters is all filters which obey the weak superposition property known as the threshold decomposition and an ordering property known as the stacking property. This class of filters includes all rank order filters, a subset of morphological filters (erosions, dilations, openings and closings), and, when both randomization of outputs and sample repetition are allowed, all linear filters.

In this paper, we show that from the class of randomizing stack filters it is possible to determine the filter which minimizes the mean absolute error between its output and a desired signal, given noisy observations of the signal as the filter's input.

1. Introduction

Stack filters are a new class of nonlinear filters that have recently been developed by several authors [1-4]. The defining properties of these new filters are two of the fundamental properties of *rank order* filters [5-7]: the threshold decomposition and the stacking property.

Because every stack filter obeys the threshold decomposition, they all have the architecture shown in Figure 1. The difference between two stack filters lies solely in the binary operation performed on each level of this architecture. As long as this operation preserves the stacking property, the filter is called a stack filter.

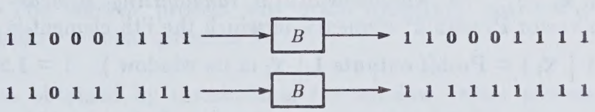


Figure 1: *A Stack Filter.* If $B(x_1,x_2,x_3) = x_2 + x_1 x_3$, the resulting stack filter S_B is the window width 3 asymmetric median filter. It preserves positive going spikes, but otherwise acts like the median filter.

In this paper, we show how to determine the stack filter which minimizes the mean absolute error between its output and the desired output signal, given noisy observations of the desired signal: the optimal window width b stack filter can be determined with a linear program with $O(b2^b)$ variables. This algorithm is very

---

This research was supported by the National Science Foundation under grant ECS83-06235.

efficient since the number of different inputs to a window width b filter is 2^b if the filter has binary input and since the number of stack filters grows faster than $2^{2^{b/2}}$.

2. The Definition of Randomizing Stack Filters

Stack filters are a subset of all window operators of a fixed window width. Defining them requires notation for the sequences that will be observed in the window of the filter. A sequence of length b will be denoted $\vec{S}_b = (S_1, S_2, \cdots, S_b)$. If the sequence is binary, lower-case letters will be used.

Two sequences $\vec{X}$ and $\vec{Y}$ of equal length, are equal if and only if $X_i = Y_i$ for all i, in which case we write $\vec{X} = \vec{Y}$. Also, $\vec{X} \leq \vec{Y}$ if $X_i = x$ implies $Y_i \geq x$; and if $\vec{X} \neq \vec{Y}$, we write $\vec{X} < \vec{Y}$.

Definition 2.1: An ordered set of sequences $\vec{S1}, \vec{S2}, \cdots, \vec{Sk}$, is said to obey the **stacking property** if

$$\vec{S1} \geq \vec{S2} \geq \cdots \geq \vec{Sk} \tag{2.1}$$

Thresholding operations create binary sequences which stack. Suppose that $\vec{S}$ is a sequence taking on M values, then define the new binary sequence $\vec{t}_\ell$, produced by thresholding $\vec{S}$ at the level ℓ, as follows:

$$t_{\ell,i} = \begin{cases} 1 & \text{if } S_i \geq \ell \\ 0 & \text{if } S_i < \ell \end{cases} \tag{2.2}$$

This thresholding operation is formally called the operator $T_\ell(\cdot)$, so that $\vec{t}_\ell = T_\ell(\vec{S})$.

Definition 2.2: A window-width b Boolean operator B: $\{0,1\}^b \rightarrow \{0,1\}$ is said to possess the stacking property if

$$B(\vec{x}) \geq B(\vec{y}) \quad \text{whenever} \quad \vec{x} \geq \vec{y} \tag{2.3}$$

A necessary and sufficient condition for B to stack is that it be a *positive Boolean function [8].*

For the purposes of the optimization procedure developed later in the paper, we must extend the definition of a positive Boolean function to allow for randomization of its output.

Definition 2.3: Let the 2^b possible binary sequences of length b be ordered in some fashion as $\vec{x}_1, \vec{x}_2, \cdots$. A window-width b *randomizing Boolean function* f($\cdot$) is defined by the vector $\vec{P}_f$ with 2^b elements, in which the i'th element is

$$P_f(1 \mid \vec{x}_i) = \text{Prob(f outputs } 1 \mid \vec{x}_i \text{ in its window)} \quad i = 1,2,\ldots,2^b \tag{2.4}$$

Definition 2.4: A randomizing Boolean function is then said to be positive if it satisfies the probabilistic stacking constraint:

$$E(f(\vec{x})) \geq E(f(\vec{y})) \quad \text{whenever} \quad \vec{x} \geq \vec{y} \tag{2.5}$$

In which E($\cdot$) is the expectation operator as defined on a suitable probability space.

Definition 2.5: Let B be a randomizing Boolean window operator which possesses the probabilistic stacking property. Then the operation of the randomizing **stack filter** $S_B(\cdot)$ applied to the M-valued sequence $\vec{S}$ is defined by

$$S_B(\vec{S}) = \sum_{i=1}^{M} B(T_i(\vec{S})) = \sum_{i=1}^{M} B(\vec{t}_i) \qquad (2.6)$$

in which $T_i(\cdot)$ is the thresholding operation defined earlier, and $\vec{t}_i$ is the i'th binary threshold signal.

3. Modeling the Noise, Signal, and Received Processes

The signal S(t), t=0,1,..., is an irreducible, discrete-time Markov Chain with state space $Q_S = \{S_1, S_2, \cdots, S_k\}$ and state transition matrix P_S. The noise N(t), t=0,1,..., is an irreducible discrete time Markov Chain with state space $Q_N = \{N_1, N_2, \cdots, N_l\}$ and state transition matrix P_N. We will assume that the signal and noise are independent.

The multi-level process which appears at the input of the filter is called the received process, R(t), t=0,1,2,.... We assume that this signal is a memoryless linear or nonlinear function of the signal and noise processes.

Since the noise and signal are independent Markov processes, the received process is the output of a memoryless function operating on a Markov process whose state space is the product space $Q_S \times Q_N$ with $m = k \cdot l$ states, and whose state transition matrix is $P_R = P_{S \times N} = P_S \otimes P_N$ ($\otimes$ is the Kronecker product.)

The Markov chain defined in the previous subsection specifies the evolution of the multi-level received process R(t) one bit at a time. There is a sequence of length b in the filter's window at any one time, though, so the process which specifies the transitions between all the possible length b window sequences must be constructed from R(t). To define the state transition matrix for this process, suppose the states in the received process state space Q_R are labeled as A,B,C,... and that there are m of them. Then the state space of the window process is $Q_W = (Q_R)^b$. List the states in Q_W as $(q_1, q_2, \cdots, q_{m^b})$ in which each q_i is a sequence of length b of states of the received process. Assume that the leftmost states in the sequences vary the fastest. Then define

$$P_W = \sum_{i=1}^{M} \vec{I}_{M,i} \otimes (\vec{I}_{M,i}^{T} P_R) \otimes \vec{I}_{M,i}^{T} \otimes I_{M^{b-2} \times M^{b-2}} \otimes \vec{I}_M \qquad (3.1)$$

in which $\vec{I}_M$ is an $M \times 1$ vector with all of its entries set to 1, $I_{M^{b-2} \times M^{b-2}}$ is an identity matrix with the specified dimension, $\vec{I}_{M,i}$ is an $M \times 1$ vector with every entry set to zero except for a 1 in the i'th position (row), and "T" is the transpose operation.

Definition 3.1: The Markov process with state transition matrix P_W is called the **multi-level window vector process** $\vec{W}(t)$.

The process obtained by thresholding the window vector process at level ℓ is called $\vec{w}_\ell(t)$, the **binary window vector process** on level ℓ. Then if the window vector at time t is $\vec{W}(t) = (W_1, W_2, \ldots, W_b)$ then the binary window vector process on level ℓ at time t has value $\vec{w}_\ell(t) = (w_1, w_2, \cdots, w_b) = T_\ell(\vec{W}(t))$.

The limiting probabilities of the multi-level window process with the state transition matrix of Equation (3.1) are arranged in a vector $\overline{\pi}$ whose entry for the i'th state of the window process is denoted as $\pi(\vec{W}_i) = \lim_{t \to \infty} P(\vec{W}(t) = \vec{W}_i)$

4. The Optimization Problem for a Single Threshold

The filters we are considering for threshold level ℓ are now formally defined.

Definition 4.1: The set $\mathcal{F}$ of window width b **randomizing binary filters** is the set of all randomizing functions $f(\cdot)$: $\{0,1\}^b \rightarrow \{0,1\}$ in which the probability that the output is a 1 or a 0 for binary vector input $\vec{x}$ is specified by the decision probability $P_f(\,1 \mid \vec{x}\,)$. The filter is thus completely specified by the vector $\vec{p}(f)$ of 2^b decision probabilities

$$\vec{p}(f) = \Big[\, P_f(\,1 \mid \vec{x}_1)\, P_f(\,1 \mid \vec{x}_2) \;\cdots\; P_f(\,1 \mid \vec{x}_{2^b}))\,\Big] \tag{4.1}$$

where $\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_{2^b}$ are all the possible binary window vectors of length b.

With the filtering functions defined, we now define a cost function for the filtering that takes place on level ℓ. The cost is based on the decisions the filter makes on each length b window vector it observes.

$C_\ell(\vec{W},0,0)$	The cost of deciding the input signal is below level ℓ when it is actually below ℓ and the (multi-level) window process is in state $\vec{W}$.
$C_\ell(\vec{W},0,1)$	The cost of deciding the input signal is equal to or greater than ℓ when it is actually below ℓ and the window process is in state $\vec{W}$.
$C_\ell(\vec{W},1,1)$	The cost of deciding the input signal is greater than or equal to ℓ when it is greater than or equal to ℓ and the window process is in state $\vec{W}$.
$C_\ell(\vec{W},1,0)$	The cost of deciding the input signal is below ℓ when it is actually greater than or equal to ℓ and the window process is in state $\vec{W}$.

The goal is then to determine the filter which minimizes the following cost.

Definition 4.2: The cost of using the randomizing binary filter f on level ℓ is a function $C(f \mid \ell)$ defined as

$$C(f \mid \ell) = \sum_{\vec{W} \in Q_W} \Big[(C_\ell(\vec{W},0,0)\pi(0 \mid \vec{W},\ell) + C_\ell(\vec{W},1,0)\pi(1 \mid \vec{W},\ell))\pi(\vec{W})\Big] P_f(\,0 \mid \vec{w}_\ell\,) + $$

$$+ \Big[(C_\ell(\vec{W},0,1)\pi(0 \mid \vec{W},\ell) + C_\ell(\vec{W},1,1)\pi(1 \mid \vec{W},\ell))\pi(\vec{W})\Big] P_f(\,1 \mid \vec{w}_\ell) \tag{4.2}$$

in which the sum is over all possible states $\vec{W}$ of the multi-level window process, $\vec{w}_\ell$ is shorthand for the binary window vector on level ℓ produced by thresholding the window process state $\vec{W}$ at level ℓ; thus, $\vec{w}_\ell = T_\ell(\vec{W})$. Also,

$$\pi(0 \mid \vec{W},\ell) = 1 - \pi(1 \mid \vec{W},\ell) = \text{Prob(correct output on level } \ell \text{ is } 0 \mid \text{state } \vec{W}). \tag{4.3}$$

which will be 0 or 1 only if complete information about the window process is available. These terms may be assigned values between 0 and 1 when the value the signal takes on in that state can only be estimated.

If the window process is binary, choosing $C(\vec{W},0,0) = C(\vec{W},1,1) = 0$, and $C(\vec{W},1,0) = C(\vec{W},0,1) = 1$ yields the absolute error criterion. This criterion is extended in Section 6 to multilevel signals using the threshold decomposition.

The cost function in Equation (4.2) takes on a simple form if we rearrange the sum, noting that several states of the multi-level window process states may produce the same binary window vector $\vec{w}$ when they are all thresholded at the same level.

$$C(f \mid \ell) = \sum_{j=0}^{2^b} \left[P_f(\,0 \mid \vec{x}_j\,) \left(\sum_{\vec{W} \in Q_W:\, \vec{w}_\ell = \vec{x}_j} \Big(C_\ell(\vec{W},0,0)\pi(0 \mid \vec{W},\ell) + C_\ell(\vec{W},1,0)\pi(1 \mid \vec{W},\ell) \Big) \pi(\vec{W}) \right) \right.$$

$$+P_f(1 \mid \vec{x}_j)\left[\sum_{\vec{W}\in Q_W:\ \vec{w}_\ell=\vec{x}_j}\left[C_\ell(\vec{W},0,1)\pi(0\mid \vec{W},\ell)+C_\ell(\vec{W},1,1)\pi(1\mid \vec{W},\ell)\right]\pi(\vec{W})\right]\Bigg] \tag{4.4}$$

The quantities in the inner parentheses in the above equation are simply

$$E(\text{Cost} \mid 0,\vec{x}_j,\ell) = E(\text{Cost} \mid 0 \text{ output } \& \ \vec{x}_j \text{ observed on level } \ell) = \tag{4.5}$$

$$\sum_{\vec{W}:\ \vec{w}_\ell=\vec{x}_j} \left(C_\ell(\vec{W},0,0)\pi(0\mid \vec{W},\ell)+C_\ell(\vec{W},1,0)\pi(1\mid \vec{W},\ell)\right)\pi(\vec{W})$$

$$E(\text{Cost} \mid 1,\vec{x}_j,\ell) = E(\text{Cost} \mid 1 \text{ output } \& \ \vec{x}_j \text{ observed onlevel } \ell) =$$

$$\sum_{\vec{W}:\ \vec{w}_\ell=\vec{x}_j} \left(C_\ell(\vec{W},0,1)\pi(0\mid \vec{W},\ell)+C_\ell(\vec{W},1,1)\pi(1\mid \vec{W},\ell)\right)\pi(\vec{W})$$

which are the expected costs of deciding zero or one for each window vector observed by the filter. With these quantities we now obtain the final form of the cost function for filtering on threshold level ℓ.

$$C(f \mid \ell) = \sum_{j=0}^{2^b}\left[P_f(0 \mid \vec{x}_j)E(\text{Cost} \mid \vec{x}_j,0,\ell)+P_f(1 \mid \vec{x}_j)E(\text{Cost} \mid \vec{x}_j,1,\ell)\right] \tag{4.6}$$

With the set $\mathcal{F}$ of randomizing binary functions and the cost function introduced above, we can state the optimization problem.

Optimization Problem: The window width b randomizing binary filter $g\in\mathcal{F}$ is said to be optimal for *window width b stack filtering* the binary threshold signal on level ℓ in the threshold decomposition architecture if it achieves

$$C(g \mid \ell) = \min_{f\in\mathcal{F}} C(f \mid \ell) \tag{4.7}$$

subject to the probabilistic stacking constraint $E(f(\vec{x})) \geq E(f(\vec{y}))$ whenever $\vec{x} \geq \vec{y}$.

5. Solving the Optimization Problem for a Single Threshold Level

5.1 Reformulation of the Probabilistic Stacking Constraint

The constraints imposed on the decision probabilities by the probabilistic stacking constraint may be seen with the aid of the unit cube in R^b. An edge of the unit cube will be denoted by the pair of vertices which it connects. Thus, if an edge connects vertices $\vec{x}$ and $\vec{y}$, the edge will be denoted $(\vec{x},\vec{y})$.

The edges of the b-cube indicate the partial ordering of the binary sequences that is created by the deterministic stacking property defined in Equation (2.1). Call any path along the edges of the b-cube an ascending path if it begins in the state 0^b (the sequence of b zeros) and proceeds "upward" through the graph to 1^b by changing exactly one 0 to a 1 at each step. For window width three, 000→001→101→111, is an ascending path. A particular ascending path in a b-cube is therefore specified by any permutation $\{k_1,k_2,\ldots,k_b\}$ of the set of integers $\{1,2,\ldots,b\}$, in which k_i specifies the location of the i'th bit to be changed from 0 to 1.

The ascending paths in a b-cube provide a complete characterization of the stacking constraints that must be satisfied by any positive Boolean function [15]. A Boolean function $f(\cdot)$ satisfies the probabilistic stacking constraint if and only if for each pair of states $\vec{x} > \vec{y}$ from the same ascending path $E(f(\vec{x})) \geq E(f(\vec{y}))$. This is the same as saying that $P_f(1 \mid \vec{x}) \geq P_f(1 \mid \vec{y})$.

For the optimality proof constructed later in the paper we need an alternate characterization of the stacking constraint in terms of the edges of the b-cube. An arbitrary edge in a b-cube will be denoted (r0s,r1s), in which r and s are taken to be arbitrary but fixed binary sequences with length L(r) and L(s), respectively. We must then have L(r)+L(s) = b-1.

Definition 5.1: A window width b Boolean function f(·) is said to satisfy the ***local probabilistic stacking constraint*** on a particular edge (r0s,r1s) in a b-cube if $E(f(r1s)) \geq E(f(r0s))$.

Lemma 5.1: A window-width b Boolean function f(·) satisfies the probabilistic stacking constraint if and only if it obeys the local probabilistic stacking constraint on every edge of the b-cube. (For proofs of the results in this paper, see [9])

At each of the two vertices of an edge (r0s,r1s) the filter may output either a 0 or a 1. Each vertex can thus be split into two pieces; for example, r0s is split into the states (r0s,0) and (r0s,1). We are then left to specify the transitions between these states to preserve the probabilistic local stacking property.

We choose these transitions to be the transitions of a Markov chain $M_{(r0s,r1s)}$ with state space {(r0s,0),(r0s,1),(r1s,0),(r1s,1)}. The transition probabilities of the Markov Chain are arbitrary, with the exception of the stacking constraint on the decisions the filter makes upon transitions between the states in the chain. We choose these transition decision probabilities to be

$$P_f(r0s \rightarrow r1s) = P(\text{transition from r0s,0 to r1s leads to r1s,1}) \quad (5.1)$$

$$P_f(r1s \rightarrow r0s) = P(\text{transition from r1s,1 to r0s leads to r0s,1})$$

then the state transition probabilities of the $M_{(r0s,r1s)}$ edge Markov chain have the values shown in Figure 2. The stacking constraint appears in these transition rates in the form of asymmetry, which can be seen in the state transition diagram.

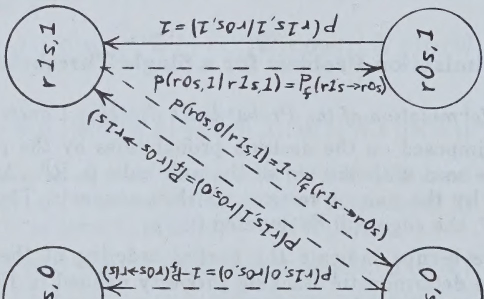

r1s,1
r0s,1
p(r1s,1/r0s,1)=1
p(r0s,1|r1s,1)=P_f(r1s→r0s)
p(r0s,0/r1s,1)=1-P_f(r1s→r0s)
p(r1s,1/r0s,0)=P_f(r0s→r1s)
p(r1s,0/r0s,0)=1-P_f(r0s→r1s)
r1s,0
r0s,0
p(r0s,0|r1s,0)=1

Figure 2: *An Edge Markov Chain.* The state transition probabilities are shown as labels on the transition arrows.

The *transition decision* probability $P_f(r1s \rightarrow r0s)$ is *not* equal to the filter decision probability $P_f(1|\, r0s)$. Defining $\pi_{(r0s,r1s)}(r0s,1)$ to be the limiting probability that the Markov Chain $M_{(r0s,r1s)}$ with the state transition diagram of Figure 4 is in the state (r0s,1), then

$$P_f(1|\, r0s) = \frac{\pi_{(r0s,r1s)}(r0s,1)}{\pi_{(r0s,r1s)}(r0s,0) + \pi_{(r0s,r1s)}(r0s,1)} = 2\pi_{(r0s,r1s)}(r0s,1) \quad (5.2)$$

The last equality follows from the choice of the transition rates, which ensure that

$$\pi_{(r0s,r1s)}(r0s,0)+\pi_{(r0s,r1s)}(r0s,1) = .5 \tag{5.3}$$

Picking the best randomizing stack filter therefore corresponds to choosing the best transition decision probabilities, $P_f(\overline{x}\rightarrow\overline{y})$ and $P_f(\overline{y}\rightarrow\overline{x})$, in each edge $(\overline{x},\overline{y})$ of the b-cube.

If the transition decision probabilities are nonzero, so that the Markov chain is irreducible, then the form of these transitions guarantees that $E(f(r1s)) \geq E(f(r0s))$, where the expected value operator is that associated with the Markov chain on the edge (r0s,r1s).

We are thus left to characterize those cases in which one or both of the decision probabilities are 0, in which case the chain defined on the state space of the edge may be reducible. To determine what occurs in this situation it is sufficient to examine the 4 possible arrangements of 0's and 1's amongst the two transition decision probabilities on a given edge. We will call the assignment of 0 or 1 to a transition decision probability a *hard* decision, and when we wish to discuss a specific pair of transition decisions they will appear in the order $[P_f(r0s\rightarrow r1s),P_f(r1s\rightarrow r0s)]$

Lemma 5.2: Suppose that both transition decisions on a given edge are hard decisions. Any combination of these decisions yields exactly one irreducible subset of 2 states whose associated outputs obey the probabilistics local stacking property with the exception of the decision [01]. The [01] decision leaves two irreducible subsets of 2 states: in one subset the output associated with both states is 0; in the other subset it is 1. Furthermore, any arrangement of filter outputs which obey the deterministic stacking property is created by at least one arrangement of hard transition decision probabilities.

The only case that requires some interpretation in the above Lemma is the [01] case. Once the cost criterion is considered, it becomes clear that if the [01] case ends up being the optimal choice of transition decision probabilities, then it must correspond to a "don't care" for the bit which is changed to create that edge.

5.2. Coordinating Decisions on Multiple Edges in the b-cube

Each edge's Markov chain is characterized by the set of equations which determine its limiting probabilities. Since there are 4 states in each edge chain, there are 4 "balance" equations relating the limiting probabilities plus the normalization equation. As above let the limiting probability for the state (r0s,0) in edge (r0s,r1s) be $\pi_{(r0s,r1s)}(r0s,0)$. Then using the state transition diagram in Figure 4, these equations are, after some rearrangement,:

$$\pi_{(r0s,r1s)}(r0s,0)+\pi_{(r0s,r1s)}(r0s,1) = .5 \tag{5.4}$$

$$\pi_{(r0s,r1s)}(r1s,0)+\pi_{(r0s,r1s)}(r1s,1) = .5$$

$$\pi_{(r0s,r1s)}(r0s,1) = P_f(r1s\rightarrow r0s)\pi_{(r0s,r1s)}(r1s,1)$$

$$\pi_{(r0s,r1s)}(r1s,1) = \pi_{(r0s,r1s)}(r0s,1)+P_f(r0s\rightarrow r1s)\pi_{(r0s,r1s)}(r0s,0)$$

For each of the $b2^{b-1}$ edges in the b-cube there is a set of edge balance equations like those above, each guaranteeing the probabilistic local stacking property for the output of the filter. Since each vertex $\overline{x}$ is an endpoint for b edges in a b-cube, the limiting probabilities for $\overline{x}$ in b of the edge Markov chains must be set equal to each other. Suppose that the vertex $\overline{x}$ is part of the b edges $(\overline{x},\overline{y}_i)$ for i = 1,2,...,b. Then we define

$$\pi(\vec{x},1) \triangleq \pi_{(\vec{x},\vec{y}_i)}(\vec{x},1) \qquad \pi(\vec{x},0) \triangleq \pi_{(\vec{x},\vec{y}_i)}(\vec{x},0) \qquad i = 1,2,\ldots,b \tag{5.5}$$

We can now define the set of all **decision constraint equations** for the b-cube. they are the minimum set of equations resulting from the union of the balance equations of each edge Markov chain with the identities specified in Equation (5.5).

The decision constraint equations:

For each vertex $\vec{x}$ *of the b-cube:*

$$\pi(\vec{x},0) + \pi(\vec{x},1) = .5 \tag{5.6}$$

For each edge (r0s,r1s) *of the b-cube:*

$$\pi(\text{r0s},1) = P_f(\text{r1s}\rightarrow\text{r0s})\pi(\text{r1s},1) \tag{5.7}$$

$$\pi(\text{r1s},1) = \pi(\text{r0s},1) + P_f(\text{r0s}\rightarrow\text{r1s})\pi(\text{r0s},0)$$

Theorem 1: The set of decision constraint equations consisting of the union over all edges of the set of edge balance equations for each edge, and the equalities of Equation (5.5), is equivalent to the probabilistic stacking constraint of Equation (2.5).

5.3. Formulation and Solution of the Optimization Problem

Equations (5.6) and (5.7) form the constraints that must be satisfied by any choice of the transition decision probabilities; that is by any randomizing stack filter. In this section, we formulate the cost criterion of Equation (4.7) in terms of the variables used in these constraint equations.

For vertex $\vec{x}_j$ of the b-cube, we have, from Equation (5.5), $P_f(1 \mid \vec{x}_j) = 2\pi(\vec{x}_j,1)$. The cost criterion of Equation (4.7) may thus be written in terms of the limiting probabilities of the edge Markov Chains as

$$C(f \mid \ell) = \sum_{j=1}^{2^b}\left[2\pi(\vec{x}_j,1)E(\text{cost} \mid \vec{x}_j) + (1-2\pi(\vec{x}_j,1))E(\text{cost} \mid \vec{x}_j)\right] \tag{5.8}$$

The cost function is then a linear function of the limiting probabilities of the vertices in the decision constraint equations. The optimization problem in Equations (5.9) and (4.8) can now be formally stated.

Optimization Problem: The randomizing filter $g \in \mathcal{F}$ is said to be optimal for stack filtering the binary threshold signal on level ℓ in the threshold decomposition architecture if it achieves

$$C(f \mid \ell) = \sum_{j=1}^{2^b}\left[2\pi(\vec{x}_j,1)E(\text{cost} \mid \vec{x}_j) + (1-2\pi(\vec{x}_j,1))E(\text{cost} \mid \vec{x}_j)\right] \tag{5.9}$$

subject to the decision constraint equations in Equations (5.6) and (5.7).

The optimization problem stated above is a linear program. The cost function and the constraints are linear functions of the limiting probabilities of the states of the all the edge Markov chains. The transition decision probabilities are handled in standard fashion [16, pg 152] by introducing slack variables. For the edge Markov chain $M_{(\text{r0s},\text{r1s})}$ define the nonnegative slack variables:

$$\nu(\text{r0s}\rightarrow\text{r1s}) = P_f(\text{r0s}\rightarrow\text{r1s})\pi(\text{r0s},0) \qquad \nu(\text{r1s}\rightarrow\text{r0s}) = P_f(\text{r1s}\rightarrow\text{r0s})\pi(\text{r1s},1) \tag{5.10}$$

There are then $2b2^{b-1}$ slack variables, each of which is nonnegative. This linear programming formulation allows us to solve the optimization problem.

Theorem 2: The optimal window width b randomizing stack filter for binary input is a window width b deterministic stack filter. The linear program to find the optimal filter has $O(b2^b)$ variables and equations.

6. Solving the Optimization Problem for Multi-level Input

In this section, we define an error criterion for filtering multi-level signals by exploiting the threshold decomposition.

Let $C(\mathbf{S}_f)$ be the expected absolute error incurred by filtering the window process $\vec{W}(t)$ with the randomizing stack filter $\mathbf{S}_f$, which is obtained by using the randomizing binary filter f on each level of the threshold decomposition architecture shown in Figure 2. Then with S(t) as the signal we are estimating

$$C(\mathbf{S}_f) = E(\mid S(t) - \mathbf{S}_f(\vec{W}(t)) \mid) = E(\mid e(t) \mid) \tag{6.1}$$

Then using the threshold function defined in Equation (2.2), and assuming, without loss of generality, that the signal S(t) and each element of the window vector process $\vec{W}(t)$ are M valued,

$$C(\mathbf{S}_f) = E(\mid \sum_{i=1}^{M} T_i(S(t)) - \sum_{i=1}^{M} T_i(\mathbf{S}_f(\vec{W}(t))) \mid) \tag{6.2}$$

in which the dimension of each argument of the threshold operation is one. Since $\mathbf{S}_f$ is a stack filter constructed from the randomizing binary filter $f \in \mathcal{F}$,

$$T_i(\mathbf{S}_f(\vec{W}(t))) = f(T_i(\vec{W}(t))) = f(\vec{w}_i(t)) \tag{6.3}$$

in which the binary vector $\vec{w}_i$ is the binary signal obtained by thresholding the multi-level window vector $\vec{W}(t)$ at level i. Equation (6.2) now becomes

$$C(\mathbf{S}_f) = E(\mid \sum_{i=1}^{M} T_i(S(t)) - \sum_{i=1}^{M} f(\vec{w}_i(t)) \mid) \tag{6.4}$$

Regroup the sum, and note that, because of the stacking property of thresholding and because of the coordination among levels of the randomizing stack filter, that all the elements of the sum are either nonnegative or nonpositive, so that the summation and the absolute value operations may be interchanged:

$$\left| \sum_{i=1}^{M} (T_i(S(t)) - f(\vec{w}_i(t))) \right| = \sum_{i=1}^{M} \left| T_i(S(t)) - f(\vec{w}_i(t)) \right| \tag{6.5}$$

which leads to

$$C(\mathbf{S}_f) = \sum_{i=1}^{M} E\left[\mid T_i(S(t)) - f(\vec{w}_i(t)) \mid \right] \tag{6.6}$$

The sum in the preceding equation is over the absolute error incurred on each threshold level. This corresponds to choosing the costs defined for filtering on level i as $C_i(\vec{W},0,0) = C_i(\vec{W},1,1) = 0$ and $C_i(\vec{W},1,0) = C_i(\vec{W},0,1) = 1$ for each threshold level i. With these costs specified in the error criterion $C(f \mid i)$ in Equation (4.2), we find that the absolute error associated with using the stack filter $\mathbf{S}_f(\cdot)$ is

$$C(\mathbf{S}_f) = \sum_{i=1}^{M} C(f \mid i) \tag{6.7}$$

the absolute error on each level is simply added together to produce the overall multi-level absolute error. Using the cost functions of the decisions of the filters for each level of the threshold decomposition architecture, we have:

$$C(\mathbf{S}_f) = \sum_{j=1}^{2^b} \left[P_f(0 \mid \vec{x}_j) \left[\sum_{i=1}^{M} E(\text{cost} \mid 0, \vec{x}_j, i) \right] + P_f(1 \mid \vec{x}_j) \left[\sum_{i=1}^{M} E(\text{cost} \mid 1, \vec{x}_j, i) \right] \right] \tag{6.8}$$

in which the costs are those experienced when making decisions on each of the M levels.

The multi-level cost function now looks exactly like the single level cost function in Equation (4.6). The optimization procedure developed in Section 5 can therefore be applied without change, yielding a positive Boolean function as the optimal filter.

Theorem 3: The optimal multi-level window width b randomizing stack filter is a deterministic stack filter; no randomization of outputs is necessary; the filter need only produce binary outputs on each level. The optimal filter can be found via a linear program with $O(b2^b)$ constraints and variables.

Proof: Exactly the same as the proof of Theorem 2.

7. Conclusion

The results derived above show the equivalence between optimal filtering under the mean absolute error criterion and massively parallel decision making under a consistency condition called the stacking property. These results also provide a theoretical framework for the absolute error criterion that is similar to the theory of optimal linear filtering under the mean-squared error criterion.

It is straightforward to extend this technique to higher dimensions and to allow multiple binary threshold signals to be fed into the filter on each threshold level. The resulting class of filters would then be a subset of the filters known as stacking morphological filters.

8. References

[1] Wendt, Coyle, and Gallagher, "Stack filters: their definition and some initial properties," *Proc. of the XIX Conf. on Information Science and Systems,* Baltimore, MD, March 1985.

[2] Wendt, Coyle, and Gallagher, "Stack filters," *IEEE Trans. on Acoustics, Speech, and Signal Processing.* vol. ASSP-34, no. 4, August 1986.

[3] Fitch, "Software and VLSI algorithms for generalized ranked order filtering," submitted to the *IEEE transactions on Circuits and Systems,* March 1986.

[4] Maragos and Schafer, "A unification of linear, median, order statistics, and morphological filters under mathematical morphology," *Proceedings of the 1985 Int. Conf. on Acoustics, Speech, and Signal Processing,* Tampa, Fla., March 1985.

[5] Fitch, Coyle, and Gallagher, "Median filtering by threshold decomposition," *IEEE Trans. Acoust., Speech, and Signal Processing,* vol. ASSP-32, pp 1183-1189, Dec. 1984.

[6] Fitch, Coyle, and Gallagher, "Threshold decomposition of multidimensional rank order operations," *IEEE Transaction on Circuits and Systems,* vol. CAS-32, pp 445-450, May 1985.

[7] Serra, *Image Analysis and Mathematical Morphology,* Academic Press: New York, 1982.

[8] Gilbert, "Lattice-theoretic properties of frontal switching switching functions," *J. Math. Physics,* vol. 33, pp 57-67, Apr. 1954.

[9] Coyle, "Choosing the stack filter which minimizes the mean absolute error criterion," submitted to *IEEE Trans. Acoust., Speech, and Sig. Proc.*

THE EFFICIENT DESIGN OF ORDER STATISTIC FILTERS

CHARLES G. BONCELET JR.
Department of Electrical Engineering
University of Delaware
Newark, Delaware 19716

ABSTRACT

We consider the classical problem of estimating signal parameters, for instance, location, with estimators based on linear combinations of order statistics. We concentrate on the design problem: how does one compute the optimal coefficients and what is the resulting mean square error. In this paper, we review some recently developed polynomial time algorithms for computing distributions of order statistics and show how they can be used to efficiently compute the optimal coefficients.

1. Introduction

We consider the classical estimation problem of Lloyd [7] and (recently popularized by) Bovik *et al.* [3] of estimating location with a linear combination of order statistics. Such estimators are no worse than simple estimates such as means and medians and are often much better. Unfortunately, the optimal design of order statistic estimators requires that one compute the covariance matrix and, depending on the optimization criteria, the mean vector of the ordered random variables. Formulae for doing this are well-known in the simple case of identically distributed observations. However, when extended to non-identically distributed random variables, these formulae have complexity which is exponential in the number of random variables. Thus, attention is often focused on small observation sizes, i.e., the number of random variables is less than or equal to five or on identically distributed random variables.

In a recent paper [2], we have presented algorithms for computing order statistic distributions which are polynomial in the number of random variables irrespective of whether or not the variables are identically distributed. In this paper, we show how to use these algorithms in designing order statistic filters. In particular, we exhibit formulae for computing means, variances, and covariances which are well matched to the recursive distribution algorithms.

We present examples illustrating the computational techniques and the efficacy of order statistic filters in comparison with other estimators.

2. Order Statistic Filters

Consider the problem of estimating an unknown parameter, θ, from a collection of observations, z_i, for $i = 1, 2, \ldots, n$. One way of doing this is to order the data from smallest to largest and form a linear combination of the ordered observations. Denote the ordered random variables by $z_{(k;n)}$, again for $k = 1, 2, \ldots, n$. (This notation is nonstandard: we are saying that $z_{(k;n)}$ is the k^{th} largest out of n random variables. Below, we will vary n.) Then the estimate can be written

$$\hat{\theta} = \sum_{k=1}^{n} \alpha_k z_{(k;n)} \tag{1}$$

with the α_k coefficients to be determined. Such an estimator is called an L-estimator. See the monograph by Huber [6] for a treatment of the theory of L-estimates as applied to robust estimation.

In 1952, Lloyd [7] considered using L-estimates to estimate location and scale from a population whose statistics depend on these two parameters. For convenience, we will reproduce Lloyd's argument: Consider $z_k = \sigma x_k + \mu$ where z_k is the observation, σ and μ are the scale and location parameters, respectively, and x_k is a normalized variate whose distribution is known. Then,

$$Ez_{(k:n)} = \sigma Ex_{(k:n)} + \mu = \sigma m_k + \mu \tag{2}$$

$$\mathrm{Var}\ z_{(k:n)} = \sigma^2 \mathrm{Var}\ x_{(k:n)} = \sigma^2 s_k^2 = \sigma^2 s_{k.k} \tag{3}$$

$$\mathrm{Cov}\ z_{(j;n)} z_{(k:n)} = \sigma^2 s_{j.k} \tag{4}$$

where m_k and $s_{j.k}$ are known because the distribution of x_i and, hence, that of $x_{(k;n)}$ is known. (Below we will discuss how computing these parameters might be accomplished.) Denote the vector of means by m, the ordered covariance matrix by S, the vector of ones by e, and the "row reversing" permutation matrix by J, i.e., for $n = 3$ one has

$$J = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}. \tag{5}$$

Under the assumption that $Jm = -m$ and $JSJ = S$, one can write the optimal least squares estimates of μ and σ as

$$\hat{\mu} = \sum_{k=1}^{n} \alpha_k z_{(k;n)} = \alpha' z \tag{6}$$

$$\hat{\sigma} = \sum_{k=1}^{n} \beta z_{(k;n)} = \beta' z \tag{7}$$

with

$$\alpha = (e'S^{-1}e)^{-1} S^{-1} e \tag{8}$$

and

$$\beta = (m'S^{-1}m)^{-1} S^{-1} m. \tag{9}$$

The variance of $\hat{\mu}$ is $\sigma^2/(e'S^{-1}e)$ and that of $\hat{\sigma}$ is $\sigma^2/(m'S^{-1}m)$.

The assumptions above hold if, as assumed by Lloyd, the x_i are independent and identically distributed with a distribution that is symmetric about 0. These assumptions also hold under the weaker condition that the *collection* of x_i is symmetrically distributed about 0. If the assumptions do not hold, then Lloyd also presents a slightly more complicated formula for α and β.

Recently, Bovik, Huang, and Munson [3] reintroduced this idea to the signal processing community under the context of "order statistic filtering" which generalized the widespread notion of median filtering [13.1]. Since that time, extensions have been proposed by several authors, among them Peterson and Kassam [11], Bovik and Naaman [4], and Palmieri and Boncelet [9,10].

3. Calculation of Order Statistic Distributions

3.1 *Background*

The principal difficulty in designing order statistic filters is in computing the covariance matrix, S and, if needed, the mean vector, m. Before one can compute these quantities, one must be able to compute the densities or the distributions of the $x_{(k:n)}$.

The theory of order statistics is well established for independent, identically distributed random variables. In the continuous case, the density of the k^{th} order statistic is formed easily:

$$f_{(k:n)}(s)ds = P(s \leq x_{(k:n)} \leq s + ds) = \frac{n!}{(k-1)!(n-k)!}F^{k-1}(s)(1-F(s))^{n-k}f(s)ds. \tag{10}$$

For discrete and mixed random variables, David and Mishriky [5] write the distribution function of $x_{(k,n)}$ as

$$P(x_{(k:n)} \leq s) = \sum_{j=k}^{n} \binom{n}{j} F^j(s)(1-F(s))^{n-j}. \tag{11}$$

For nonidentically distributed and dependent random variables, the theory of order statistics is not as well understood or as convenient as that for the independent, identically distributed case. Vaughan and Venables [12] provide a permanent expansion of the density for continuous, independent, nonidentically distributed random variables. For the application to signal filtering Ataman *et al.* [13] studied the properties of the median for the $n = 3$ case. Maurer and Margolin [15], using the inclusion-exclusion formula, present an expression for the distribution of order statistics for dependent random variables. Recently, Nodes and Gallagher [8] considered the median filtering problem and presented expressions for the distributions of order statistics with various overlapping windows. The objection to these generalizations is that they are all *exponential* algorithms, that is, the number of operations needed is an exponential function of the number of variables, n.

3.2. *Recursive Formulae for Independent Random Variables*

In a recent paper [2], we have presented a series of recursive algorithms to compute order statistic distributions when the random variables are either independent or Markov. No assumptions are necessary regarding the distributions: The random variables may be discrete, mixed, or continuous. They may be identically distributed or not. In this paper, we will detail these methods for calculating first and second moments.

We make the following definitions: as above, denote the k^{th} order statistic out of n random variables by $x_{(k;n)}$. For first order distributions, let $-\infty = s_0 < s_1 < s_2 = +\infty$ and, for second order, let $-\infty = s_0 < s_1 < s_2 < s_3 = +\infty$ and let $k_1 \leq k_2$. Then we have the following theorem on events of order statistics:

$$\{x_{(k;n+1)} \leq s_1\} = \{x_{(k;n)} \leq s_1\}\{x_{n+1} > s_1\} + \{x_{(k-1;n)} \leq s_1\}\{x_{n+1} \leq s_1\} \tag{12}$$

$$\begin{aligned}\{x_{(k_1;n+1)} \leq s_1, x_{(k_2;n+1)} \leq s_2\} &\\ = \{x_{(k_1;n)} \leq s_1, x_{(k_2;n)} \leq s_2\}&\{x_{n+1} > s_2\} \\ + \{x_{(k_1;n)} \leq s_1, x_{(k_2-1;n)} \leq s_2\}&\{s_1 < x_{n+1} \leq s_2\} \\ + \{x_{(k_1-1;n)} \leq s_1, x_{(k_2-1;n)} \leq s_2\}&\{x_{n+1} \leq s_1\}\end{aligned} \tag{13}$$

In the first order case, this theorem simply states that there are two ways the k^{th} order statistic out of $n + 1$ random variables can be $\leq s_1$: one, the $n + 1^{st}$ is larger than s_1 and the k^{th} order statistic out of n is $\leq s_1$ and two, the $n + 1^{st}$ is $\leq s_1$ and the $k - 1^{st}$ order statistic out of n is $\leq s_1$. In the second order case, the event in question is similarly decomposed into three events.

Notice that the events on the right have side are disjoint since the events on x_{n+1} partition the real line into nonoverlapping segments. A direct corollary of this theorem is a recursive formula for calculating distributions for independent x_i:

$$P(x_{(k;n+1)} \leq s_1) = P(x_{(k;n)} \leq s_1)(1 - F_{n+1}(s_1)) + P(x_{(k-1;n)} \leq s_1)F_{n+1}(s_1) \quad (14)$$

$$\begin{aligned} P(x_{(k_1;n+1)} \leq s_1, x_{(k_2;n+1)} \leq s_2) \\ = P(x_{(k_1;n)} \leq s_1, x_{(k_2;n)} \leq s_2)(1 - F_{n+1}(s_2)) \\ + P(x_{(k_1;n)} \leq s_1, x_{(k_2-1;n)} \leq s_2)(F_{n+1}(s_2) - F_{n+1}(s_1)) \\ + P(x_{(k_1-1;n)} \leq s_1, x_{(k_2-1;n)} \leq s_2)F_{n+1}(s_1) \end{aligned} \quad (15)$$

3.3. *Recursive Formulae for Markov Random Variables*

The formulae above for propagating the distributions can be extended to Markov random variables. It will be convenient to reverse the time index on the Markov process so that it is propagating backward in time and it will also be convenient to consider only a discrete Markov process (so that integrals can be replaced by sums). The idea is to condition on knowing x_n so that the events become independent (one is dependent on x_i for $i = 1, 2, \ldots, n$; the other on x_{n+1}) and then to integrate out x_n. For details, see [2].

The formulae for computing distributions of discrete, Markov random variables come in two parts, the first being simply the law of total probability:

$$P(x_{(k;n+1)} \leq s_1) = \sum_{t_{n+1}} P(x_{(k;n+1)} \leq s_1 | x_{n+1} = t_{n+1})P(x_{n+1} = t_{n+1}) \quad (16)$$

$$\begin{aligned} P(x_{(k_1;n+1)} \leq s_1, x_{(k_2;n+1)} \leq s_2) = \\ \sum_{t_{n+1}} P(x_{(k;n+1)} \leq s_1, x_{(k_2;n+1)} \leq s_2 | x_{n+1} = t_{n+1})P(x_{n+1} = t_{n+1}) \end{aligned} \quad (17)$$

The second part follows from the first theorem and the discussion above:

$$\begin{aligned} P(x_{(k;n+1)} \leq s_1 | x_{n+1} = t_{n+1}) \\ = \sum_{t_n} P(x_n = t_n | x_{n+1} = t_{n+1}) \Big(I^1_{n+1} P(x_{(k-1;n)} \leq s_1 | x_n = t_n) \\ + I^2_{n+1} P(x_{(k;n)} \leq s_1 | x_n = t_n) \Big) \end{aligned} \quad (18)$$

$$\begin{aligned} P(x_{(k_1;n+1)} \leq s_1, x_{(k_2;n+1)} \leq s_2 | x_{n+1} = t_{n+1}) \\ = \sum_{t_n} P(x_n = t_n | x_{n+1} = t_{n+1}) \\ \cdot \Big(I^1_{n+1} P(x_{(k_1-1;n)} \leq s_1, x_{(k_2-1;n)} \leq s_2 | x_n = t_n) \\ + I^2_{n+1} P(x_{(k_1;n)} \leq s_1, x_{(k_2-1;n)} \leq s_2 | x_n = t_n) \\ + I^3_{n+1} P(x_{(k_1;n)} \leq s_1, x_{(k_2;n)} \leq s_2 | x_n = t_n) \Big) \end{aligned} \quad (19)$$

where $I^j_{n+1} = 1$ if $s_{j-1} < t_{n-1} \le s_j$ and $I^j_{n+1} = 0$ otherwise.

We have also extended these results to other cases, for example, to a Markov sequence corrupted by independent noise, but these generalizations will be presented elsewhere.

3.4. *Discussion*

The algorithms presented above have polynomial complexity. In particular, each application of the second order formula for independent variates with fixed s_1 and s_2 requires $O(n^3)$ operations. If we assume that there are $O(S^2)$ values for (s_1, s_2), then this formula requires $O(S^2 n^3)$ operations to calculate all the points. The Markov case is worse since we must calculate $O(S^3)$ quantities with each requiring $O(S)$ work for each combination of k_1 and k_2, yielding a total complexity of $O(S^4 n^2)$ operations.

Individual points of the distributions are calculated independently of all others. This property makes the algorithm ideally suited for a recursive implementation. It also means that one should structure calls so that the points, (s_1, s_2), vary slowly, i.e., so that one calculates the same point for each combination of (k_1, k_2) before changing (s_1, s_2).

4. Moment Calculation

The algorithm discussed above calculates distribution values, not density values. The usual moment formulae involve integrating against the density. For discrete random variables, this difference is minor because the distribution can be "differenced" to obtain the density. However, differencing the distribution can incur a performance penalty since distribution values for different points have to be stored. Further, for continuous random variables, the distribution must be differentiated to get the density. This is not possible when the distribution is calculated numerically, so some approximation must be made.

The approach we suggest is to quantize the random variables and calculate the moments of the quantized random variables. Below, we present formulae for the exact calculation of moments of discrete random variables which do not require that distribution be differenced. This will mesh well with the distribution formulae above.

We are also guaranteed that the moments of the discrete random variables will converge to those of the continuous ones (assuming the moments are finite) because the distributions of the ordered discrete variates are equal to those of the ordered continuous variates at each breakpoint. That being the case, proving convergence is a simple Lebesque dominated convergence argument. However, at this time we do not have a theory which indicates the rate of convergence. In this paper, we have relied on numerical experiments to establish convergence.

Consider the random variables to be discrete and defined on s_i, for $i = 1, 2, \ldots, S$. Let $a_i \equiv A(s_i)$, $p_i \equiv P(x_{(k:n)} = s_i)$, and $C_i \equiv P(x_{(k:n)} \le s_i)$. C satisfies the boundary conditions, $C_0 = 0$ and $C_S = 1$. Then a formula for first order moments can be found from the "summation by parts" formula [14]:

$$\sum_{i=1}^{S} a_i (C_i - C_{i-1}) = a_S C_S - \sum_{i=1}^{S-1} (a_{i+1} - a_i) C_i - a_1 C_0 \tag{20}$$

Thus.

$$EA(x_{(k;N)}) = \sum_{i=1}^{S} a_i p_i = \sum_{i=1}^{S} a_i (C_i - C_{i-1}) \tag{21}$$

$$= a_S - \sum_{i=1}^{S-1} (a_{i-1} - a_i) C_i \tag{22}$$

When $s_i = \gamma i - \delta$. particularly simple formulas result for the first and second moments:

$$Ex_{(k;N)} = \gamma S + \delta - \gamma \sum_{i=1}^{S-1} C_i \tag{23}$$

$$Ex^2_{(k;N)} = (\gamma S + \delta)^2 - \gamma \sum_{i=1}^{S} (2\gamma i - 2\delta - \gamma) C_i$$

Second order moments of order statistics are a little more difficult due to the dependence of the two order statistics. In particular. for $k_2 > k_1$. $x_{(k_2;N)} \geq x_{(k_1;N)}$. As above, let

$$a_{i,j} \equiv A(s_i, s_j) \tag{24}$$

$$p_{i,j} \equiv P(\{x_{(k_1;N)} = s_i\}, \{x_{(k_2;N)} = s_j\}) \tag{25}$$

$$C_{i,j} \equiv F(k_1, k_2; N)(s_i, s_j) \tag{26}$$

Then $p_{i,j} = C_{i,j} - C_{i-1,j} - C_{i,j-1} + C_{i-1,j-1}$. The $C_{i,j}$ satisfy the following boundary conditions: $C_{0,j} = C_{i,0} = C_{0,0} = 0$, $C_{S,S} = 1$, and $C_{i,j} = C_{j,j}$ for $i > j$.

We will summarize the two dimensional formula in terms of a lemma.

Lemma 1:

$$EA(x_{(k_1;N)}, x_{(k_2;N)}) = a_{S,S} + \sum_{j=2}^{S-1} \sum_{i=1}^{j-1} (a_{i+1,j+1} - a_{i,j+1} - a_{i+1,j} + a_{i,j}) C_{i,j}$$

$$- \sum_{i=1}^{S-1} (a_{i+1,S} - a_{i,S}) C_{i,S} - \sum_{j=1}^{S-1} (a_{j,j+1} - a_{j,j}) C_{j,j} \tag{27}$$

Proof: See Boncelet [2].

Although these formulae appear complicated, they are not difficult to implement. In particular, each C_i and $C_{i,j}$ is used only once in each formula. We never have to store these values. Of course in some applications, the distributions may possess symmetry which can be exploited, so some storage may be warranted. Further, these formulae make no restriction on the s_i. In particular, we are not assuming a uniform grid.

5. Conclusion

In this paper, we have been concerned with the *design* of order statistic filters and, in particular. how to do the design in an efficient fashion. Toward this end. we have surveyed order statistic filters. presented formulae for computing distributions of order statistics, and presented formulae for computing moments of discrete ordered random variables. We have not been concerned with the matrix inversion nor with implementing an order statistic filter. The matrix inversion requires comparatively little effort when compared to that required for computing the matrix. Implementing an order statistic filter requires that data to be sorted, then a dot product of the sorted data and the coefficients. These are relatively simple operations.

The formulae presented here are believed to be the fastest available. Hopefully, this paper will encourage others to explore this estimators, especially in nonidentically distributed situations.

References

1. G. Arce and N. Gallagher, Jr., "State description for the root-signal set of median filters." *IEEE Trans. on Acoust. Speech. Signal Proc.*, vol. ASSP-30, no. 6, pp. 894-902. Dec. 1982.
2. C. G. Boncelet, Jr., *Algorithms to compute order statistic distributions.* To appear in SIAM J. on Sci. and Stat. Computing
3. A. C. Bovik, T. S. Huang, and D. C. Munson. Jr., "A generalization of median filtering using linear combinations or order statistics." *IEEE Trans. on Acoust., Speech, Signal Proc.*, vol. ASSP-31, no. 6, pp. 1342-1350. Dec. 1983.
4. A. C. Bovik and L. Naaman, "Least-squares signal estimation using order statistic filters," in *Proceedings of the 1986 Conference on Information Sciences and Systems*, pp. 735-739, Princeton University. Princeton. NJ, March, 1986.
5. H. A. David and R. S. Mishriky, "Order statistics for discrete populations and for grouped samples," *J. Amer. Statist. Assoc.*, vol. 63, p. 1390, 1968.
6. P. J. Huber, *Robust Statistics,* J. Wiley & Sons, New York, NY. 1981.
7. E. H. Lloyd, "Least-squares estimations of location and scale parameters using order statistics." *Biometrika*, vol. 39, pp. 88-95, 1952.
8. T. A. Nodes and N. C. Gallagher, Jr., "The output distribution of median type filters." *IEEE Trans. on Commun.*, vol. COM-32, no. 5, pp. 532-541, May, 1984.
9. F. Palmieri and C. G. Boncelet, "Optimal MSE linear combination of order statistics for restoration of Markov processes," in *Proceedings of the 1986 Conference on Information Sciences and Systems*, pp. 729-734, Princeton University. Princeton, NJ, March, 1986.
10. F. Palmieri and C. G. Boncelet, "Ll-Filters," in *Proceedings of the Twenty-Fourth Annual Allerton Conference on Communication, Control, and Computing*, University of Illinois, Urbana, IL, October, 1986.
11. S. R. Peterson and S. A. Kassam, "General order statistic-based structures for nonlinear signal restoration," in *Proceedings of the 1986 Conference on Information Sciences and Systems*, pp. 723-728, Princeton University, Princeton, NJ, March, 1986.
12. R. J. Vaughan and W. N. Venables, "Permanent expressions for order statistic densities," *J. Roy. Statist. Soc.*, vol. 34, pp. 308-310, 1972.
13. E. Ataman, V. K. Aatre, and K. M. Wong, "Some statistical properties of median filters," *IEEE Trans. on Acoust., Speech, Signal Processing*, vol. ASSP-29, no. 5, pp. 1073-1075, Oct., 1981.
14. D. E. Knuth, *Fundamental Algorithms,* Addison-Wesley, 1975.
15. W. Maurer and B. H. Margolin, "The multivariate inclusion-exclusion formula and order statistics from dependent variates," *The Annals of Statistics*, vol. 4, no. 6, pp. 1190-1199, 1976.

THE EQUIVALENCE OF ORDER STATISTIC AND NONRECURSIVE LINEAR DIGITAL FILTERS UNDER MONOTONICITY CRITERIA

HAROLD G. LONGBOTHAM and ALAN C. BOVIK
Department of Electrical & Computer Engineering
The University of Texas at Austin
Austin, TX 78712-1084

Abstract -- While there exist signal processing domains where order statistic filters (OS filters) afford advantages over linear filtering strategies, there are many more situations where the converse holds, or where the filters may produce similar results. Before a methodology for designing OS filters can be developed, these application domains must be delineated. This implies that the differences resulting from linear and OS filtering of a given class of signals must be quantified, and conversely, those signal classes which would benefit from OS filtering as opposed to linear filtering must be identified. A first approach to this problem involves finding those signals for which OS and linear filtering produces identical results, when the filters are defined with identical coefficients; this is motivated by the fact that linear and OS filters are intimately related, as OS filtering can be regarded as linear filtering modified by a nonlinear ordering operation. In this paper, we derive necessary and sufficient conditions on the signal process for nonrecursive OS and linear filters to result in identical outputs. This work extends previous theories advanced by Gallagher and Wise [6] and Tyan [7] characterizing the well-known median filter.

I. INTRODUCTION

The median filter was first proposed by J.W. Tukey in 1971 [1] as a smoothing device for discrete signals. In particular, he noted that the filter was effective for suppressing impulse noise while simultaneously preserving edge-type signal structures containing significant information. These properties made the median filter a natural choice for noise reduction purposes in many problems of practical interest; over the next ten years many papers appeared which demonstrated the efficacy of the filter for applications such as speech [2] and image processing [3], and fast algorithms were developed to accelerate its computation [4,5]. However, despite these advances and the apparent value of the technique in many applications, the median filter was widely regarded as a curiosity with no solid foundation in signal processing theory. This situation changed in 1981 when Gallagher and Wise [6] and Tyan [7] independently developed similar theories describing the deterministic properties of the filter; in particular, it was shown that signals containing a minimum degree of local monotonicity are invariant to median filtering (labeled *root signals* in [6]), and that repeated application of the filter to a signal results in a root within a finite number of filter passes.

These results gave the median filter a theoretical groundwork, and spurred the development of a number of extensions and generalizations, including *ranked-order filters*, where a single order statistic (generally other than the median) from the windowed data set is reproduced at each signal coordinate [8], and the more general *order statistic filters* (OS filters or *L-filters*) [9,10], where a linear combination of the windowed order statistics are taken as the filter output at each coordinate. OS filters (and modifications discussed in [10]) are interesting because (a) they offer a compromise in performance between linear filtering and median filtering, and (b) it is possible to design an optimal (among OS filters) MSE filter for estimating a constant signal level immersed in any iid noise, whose performance will be superior to linear filtering.

More recently, morphological filters [11] and stack filters [12] have been shown to be extensions of OS filters. Morphological filters include both OS filters and certain linear filters as special cases, while stack filters include both linear and morphological filters if operating on quantized signals. Coyle [13] has developed means for finding the optimal stack filter under an absolute error criterion, and others have found means for computing the optimal MSE OS filter coefficients for estimating nonconstant signals in iid noise [14] and for restoration of Markov sequences [15]. Efficient VLSI implementations for stack filters have also been suggested, and a chip has been built which can perform any rank order operation of window width nine or less [12]. Thus, the concept of median filtering as an ad hoc signal processing tool has begun to metamorphose into a rich theory for signal estimation/restoration which provides significant advantages over linear filtering methodologies, both in performance and in implementation.

Yet, many questions regarding the applicability of these filters to more general signal processing applications remain unanswered. The most generic goal of a digital filter is to enhance or preserve some component of an input signal, while suppressing some other component. The Fourier interpretation of frequency selection has made linear filtering an irreplaceable tool for manipulating signals; e.g., there is no clearly defined theory for designing an optimal "high-pass" type nonlinear filter under any criterion. While there clearly exist applications where median-type filters (here, "median-type" refers to stack, morphological, ranked-order, or OS filters) afford an advantage over linear filtering strategies, it must also be realized that there are many more situations where the converse holds, or where the filters may produce identical or similar results. Before a concrete methodology can be advanced for designing median-type filters as an alternative to linear filtering, these application domains must be delineated, from both a deterministic and statistical point of view.

In this paper, we address the deterministic problem for OS filters only, for the simple reason that OS and linear filters are intimately related: OS filters may be regarded as modifications of nonrecursive linear filters (windowing followed by linear weighting) via the inclusion of a nonlinear ordering operation between windowing and weighting. The problem then may be restated as: For what signal processing applications does the inclusion of an ordering element (prior to coefficient selection) produce a benefit? Unfortunately, the answer to this question relies upon some knowledge of the particular application being considered. More generally we might ask: How do the outputs resulting from OS and linear filtering of a signal relate? A first approach to this question involves finding those signals for which OS and linear filtering produce identical results, when the filters are defined with identical coefficients, viz, identifying those signals for which inclusion of the ordering element produces no difference at the output. The papers by Gallagher and Wise [6], Tyan [7], and Nodes and Gallagher [8] provide a clue to determining what these signals are; it turns out that the signal must either be monotonic or locally monotonic. However, there are certain conditions that the filter coefficients must also satisfy; interestingly, the linear filters satisfying these conditions are those (within an additive constant) which can be represented as morphological filters.

II. RANKED-ORDER FILTERS AS TIME DELAYS ON MONOTONIC SIGNALS

First we introduce the notation used throughout the paper.

A. Notation

$\{x_i\}$ - Two-sided infinite finite-valued data sequence to be filtered.

$\{\mathbf{x}_i\}$ - Sequence of windowed data vectors $\mathbf{x}_i = (x_{1:i}, x_{2:i}, \ldots, x_{2N+1:i})^T$, where $x_{j:i} = x_{i+j-N-1}$, so that at each signal coordinate i, the filter is centered at i. Unless otherwise stated, the filter span is taken to be 2N+1, where N > 0.

$\{\mathbf{x}_{(i)}\}$ - Sequence of ordered data vectors $\mathbf{x}_{(i)} = \text{order}\{\mathbf{x}_i\} = (x_{(1):i}, x_{(2):i}, \ldots, x_{(2N+1):i})^T$, so that $x_{(j):i} \le x_{(k):i}$ whenever $j \le k$.

α - Coefficient vector $(\alpha_1, \alpha_2, \ldots, \alpha_{2N+1})^T$ associated with an arbitrary OS or linear filter.

β - Coefficient vector $(\beta_1, \beta_2, \ldots, \beta_{N+1}, \ldots, \beta_2, \beta_1)^T$ associated with an arbitrary *symmetric* OS or linear filter.

$\mathbf{1}$ - Constant-valued vector $(1, 1, \ldots, 1)^T$ of length 2N+1.

$\{L_\alpha(\mathbf{x}_i)\}$ - OS-filtered sequence with elements $L_\alpha(\mathbf{x}_i) = \alpha^T \mathbf{x}_{(i)}$.

$\{F_\alpha(\mathbf{x}_i)\}$ - Linear-filtered sequence with elements $F_\alpha(\mathbf{x}_i) = \alpha^T \mathbf{x}_i$.

By assuming the input sequence to be two-sided infinite, we are following Tyan [7], and will require similar conditions to those used by him (the assumption that least one monotonic region is present in the input signal). However, the theories given here could also be proven under conditions similar to those used by Gallagher and Wise in [6] (finite-length input signals with padding at both ends).

B. Ranked-Order Filters

By a *ranked-order filter,* we mean an OS filter with all coefficients zero except for one, which we set to unity, viz, $\alpha_j = 0$ for $j = 1, \ldots, 2N+1$, $j \ne k$, and $\alpha_k = 1$ for some $k \in \{1, \ldots, 2N+1\}$. Hereafter, we will refer to such a filter as a *kth ranked-order filter.* Using our previous notation, we may express the output of a kth ranked-order filter as $L_\alpha(\mathbf{x}_i) = x_{(k):i}$, so that the *median filter* is defined as $L_\alpha(\mathbf{x}_i) = x_{(N+1):i}$. Nodes and Gallagher [8] discussed many properties of ranked-order filters. In particular, they found that an ideal (step) edge will be shifted (k-N-1) units following filtering with a kth ranked-order filter, where the edge is advanced if the edge is positive-going, and delayed if the edge is negative-going. This is precisely the result obtained by the linear filter $F_\alpha(x_i) = x_{k:i}$ if the edge is positive-going, and by $F_\alpha(x_i) = x_{2N+2-k:i}$ if the edge is negative-going. If we further note that ideal step edges are either monotone nondecreasing or monotone nonincreasing, we may hypothesize that kth ranked-order filters and linear time delay units are equivalent when operating on monotone signals, with special provision being made for the median filter. These ideas are formalized in the following.

Theorem 1: *Suppose $\{x_i\}$ contains at least one nondecreasing segment $(x_p, \ldots, x_{p+2N})$ of length 2N+1. Then the following two statements are equivalent:*

(1) $\{x_i\}$ is everywhere nondecreasing.

(2) $\{x_{(k):i}\} = \{x_{k:i}\}$ for any $k \in \{1, \ldots, 2N+1\}$, $k \neq N+1$.

The following Lemma will be used in proving Theorem 1.

Lemma 1: *Suppose $\{x_i\}$ contains at least one nondecreasing segment $(x_p, \ldots, x_{p+2N})$ of length 2N+1. Then the following two statements are equivalent:*

(1) $\{x_i : p \leq i < \infty\}$ is nondecreasing if and only if $\{x_{(k):i} : p+N \leq i < \infty\} = \{x_{k:i} : p+N \leq i < \infty\}$ for any $k \in \{1, 2, \ldots, 2N+1\}$, $k \neq N+1$.

(2) $\{x_i : -\infty < i \leq p\}$ is nondecreasing if and only if $\{x_{(k):i} : -\infty < i \leq p+N\} = \{x_{k:i} : -\infty < i \leq p+N\}$ for every $k \in \{1, 2, \ldots, 2N+1\}$, $k \neq N+1$.

Proof: Without loss of generality we may take $p = -N$. Assume that (1) is true, viz, that $\{x_i : -N \leq i < \infty\}$ is nondecreasing if and only if $\{x_{(k):i} : 0 \leq i < \infty\} = \{x_{k:i} : 0 \leq i < \infty\}$ for every $k \in \{1, \ldots, 2N+1\}$, $k \neq N+1$. We will now show that this must imply that (2) is true as well.

Since $\{x_i : -N \leq i \leq N\}$ is nondecreasing, $\{-x_{-i} : -N \leq i \leq N\}$ is as well. By assumption we have that $\{-x_{-i} : -N \leq i < \infty\}$ is nondecreasing if and only if $\{-x_{(k):-i} : 0 \leq i < \infty\} = \{-x_{k:-i} : 0 \leq i < \infty\}$ for every $k \in \{1, \ldots, 2N+1\}$, $k \neq N+1$. Since $\{x_i : -\infty < i \leq N\}$ is nondecreasing if and only if $\{-x_{-i} : -N \leq i < \infty\}$ is nondecreasing, and since $\{-x_{(k):-i} : 0 \leq i < \infty\} = \{-x_{k:-i} : 0 \leq i < \infty\}$ for every $k \in \{1, \ldots, 2N+1\}$, $k \neq N+1$, if and only if $\{x_{(k):i} : -\infty < i \leq 0\} = \{x_{k:i} : -\infty < i \leq 0\}$ for every $k \in \{1, \ldots, 2N+1\}$, $k \neq N+1$, we have that $\{x_i : -\infty < i \leq -N\}$ is nondecreasing if and only if $\{x_{(k):i} : -\infty < i \leq 0\} = \{x_{k:i} : -\infty < i \leq 0\}$ for any $k \in \{1, \ldots, 2N+1\}$, $k \neq N+1$.

The proof that (2) implies (1) is identical. ♦♦

Proof (Theorem 1): That (1) implies (2) is obvious, since the rank ordering coincides with the ordering of the signal indices.

To prove that (2) implies (1), we first note that the proof is obvious if k = 1 or k = 2N+1 (since $x_{(1):p+n-1} = x_{1:p+n-1}$ implies that $x_{p-1} \leq x_p$, and $x_{(2N+1):p+n+1} = x_{2N+1:p+n+1}$ implies that $x_{p+2N} \leq x_{p+2N+1}$, and by using Lemma 1).

Suppose then that $k \in \{2, \ldots, 2N\}$, $k \neq N+1$. By contradiction let us assume that $\{x_i\}$ is not everywhere nondecreasing; using Lemma 1, we may assume that monotonicity is violated at some signal coordinate larger than p+2N. Suppose then that j is the least integer such that $j > p+2N$ and $x_{k:j+N+1} > x_{k:j+N+2}$ (equivalently, $x_{j+k} > x_{j+k+1}$). Without loss of generality we may take j=0. Then by hypothesis we have $x_1 \leq x_2 \leq \ldots \leq x_k > x_{k+1}$ and $x_{(k):N+1} = x_k$ $(= x_{k:N+1})$, and therefore $x_{k-1} = x_k > x_{k+1}$. Since $x_{(k):N} = x_{k:N}$ (equivalently, $x_{(k-1):N+1} = x_{k-1}$) we see that we must similarly have $x_{k-2} = x_{k-1}$. Continuing in this manner we find that the 2N+1-k elements preceding x_k must all be equal to x_k.

Suppose now that $N+2 \leq k \leq 2N$, and let m = k-N-1. By hypothesis we have $x_2 \leq \ldots \leq x_{2m+1} = \ldots = x_k > x_{k+1}$ and $x_{(k):N+2} = x_{k+1}$ $(= x_{k+1:N+1})$. Since $0 < m < N$ we see that $x_{2m+1} \in x_{N+2}$ for any k satisfying $N+2 \leq k \leq 2N$, there must be N-m elements immediately preceding x_k which are equal to x_k. But if $x_{k+1} = x_{(k):N+2}$ there can be at most N-m elements of x_{N+2} which are strictly greater than x_{k+1}, which is a contradiction.

Now let us suppose that $2 \le k \le N$. In order to have $x_{k:N+1}$ (= x_k) = $x_{(k):N+1}$ we must have $x_1 = \ldots = x_k > x_{k+1}$, since the 2N+1-k elements preceding x_k are equal to x_k. Similarly, if $x_{k:N+2}$ (= x_{k+1}) = $x_{(k):N+2}$ there must be at least k elements of $S = \{x_{k+1}, \ldots, x_{2N+2}\}$ that are not greater than x_{k+1}. Therefore, there must be exactly k elements in S that are strictly less than x_k, and one of these elements must be x_{2N+2}. Let $n = \min\{i : x_i \in S \text{ and } x_i \ge x_k\}$. Then $k+2 \le n \le 2k$ since $x_n \notin \{x_{k+1}, x_{2N+2}\}$. Furthermore, we have $x_n = x_{k:n-k+N+1}$, so that $x_{1:n-k+N+1} = x_q$ where q = n-k+1. Then $3 \le q \le k+1$, so that the k elements of S that are strictly less than x_k all lie in $x_{n-k+N+1}$. Since these elements are also strictly less than x_n we have that $x_n \ne x_{(k):n-k+N+1}$, which is a contradiction. ◆◆

In the preceding proof, it should be noted that contradictions are not reached if k = N+1 (the median filter) were allowed.

Corollary 1.1: *Suppose $\{x_i\}$ contains at least one nonincreasing segment $(x_p, \ldots, x_{p+2N})$ of length 2N+1. Then the following two statements are equivalent:*

(1) $\{x_i\}$ is everywhere nonincreasing.

(2) $\{x_{(k):i}\} = \{x_{2N+2-k:i}\}$ for any $k \in \{1, \ldots, 2N+1\}$, $k \ne N+1$.

Proof: We need only note that the sequence $\{x_i\}$ is everywhere nonincreasing if and only if $\{x_{-i}\}$ is everywhere nondecreasing and by using Theorem 1. ◆◆

Theorem 1 and its corollary do not include the median filter, for the simple reason that they do not hold for k = N+1. However, under different monotonicity conditions (local monotonicity), median filtering can be shown to be equivalent to linear identity filtering, as given in the following restatement of a theorem due to Tyan [7].

Definition: A sequence $\{x_i\}$ is *locally monotonic of order n*, or *LOMO(n)*, if $\{x_j, \ldots, x_{j+n-1}\}$ is monotonic (nonincreasing or nondecreasing) for every j. A LOMO(n) sequence is also LOMO(p) whenever p < n.

Theorem 2: *Suppose that $\{x_i\}$ contains at least one monotonic (nonincreasing or nondecreasing) segment $(x_p, \ldots, x_{p+N})$ of length N+1. Then $\{x_{(N+1):i}\} = \{x_{N+1:i}\}$ if and only if $\{x_i\}$ is LOMO(N+2).*

Proof: See [7]. ◆◆

We have established that the subclass of OS filters known as ranked-order filters are equivalent to linear time-delay filters when operating on signals which are sufficiently smooth in the sense that they are monotonic, and that they yield different results otherwise, provided that the signal contains at least one smooth (monotonic) region. The exception to this is the median filter, which is the only ranked-order filter whose coefficients are symmetrically defined. Median filtering is the same as linear identity filtering when operating on signals which are locally monotonic of sufficient order. We may ask the more general question: Under what conditions on the signal $\{x_i\}$ do we have $\{L_\alpha(x_i)\} = \{F_\alpha(x_i)\}$ for an arbitrary filter coefficient set α? We address this question in the following section.

III. EQUIVALENCE OF NONRECURSIVE OS AND LINEAR FILTERING OF LOMO SIGNALS

A. Filters Defined with Arbitrary Coefficients

Finding necessary and sufficient conditions on the input signal $\{x_i\}$ and the filter coefficient vector α such that $\{L_\alpha(x_i)\} = \{F_\alpha(x_i)\}$ is, in general, a very difficult problem. The problem is complicated by the fact that in taking a linear combination of the ordered data samples at any signal coordinate i, the output may be the same as that obtained by a linear weighting of the unordered samples, even if the order statistics and the original unordered data samples do not coincide at any point, viz, $x_{(k):i} \ne x_{k:i}$ for k = 1, . . . 2N+1. However, we may show that when $\{L_\alpha(x_i)\} = \{F_\alpha(x_i)\}$ for *any* coefficient vector α, or for *any* input signal $\{x_i\}$, that certain conditions are induced on the signal and filter coefficients, respectively. We will make extensive use of the following Lemma:

Lemma 2: *Suppose that* $\mathbf{p} = p\mathbf{1}$ *where* p *is any real scalar constant, and let* $q \neq 0$ *be any other real scalar constant. Further, suppose that* α *is any arbitrary filter coefficient set, and* β *is any arbitrary symmetric filter coefficient set. Then,* $\{L_\alpha(x_i)\} = \{F_\alpha(x_i)\}$ *if and only if each of the following hold:*

(a) $\{L_{\alpha+\mathbf{p}}(x_i)\} = \{F_{\alpha+\mathbf{p}}(x_i)\}$

(b) $\{L_{q\alpha}(x_i)\} = \{F_{q\alpha}(x_i)\}$

(c) $\{L_\alpha(x_i+\mathbf{p})\} = \{F_\alpha(x_i+\mathbf{p})\}$.

Further, $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ *if and only if the following holds:*

(d) $\{L_\beta(qx_i)\} = \{F_\beta(qx_i)\}$.

Proof: Items (a)-(c) may be easily seen to be true by noting that we may write $L_\alpha(\mathbf{x}_i) = \boldsymbol{\alpha}\cdot\mathbf{x}_{(i)}$ and $F_\alpha(\mathbf{x}_i) = \boldsymbol{\alpha}\cdot\mathbf{x}_i$, where '·' is the usual vector inner product. Item (d) may be seen to be true for $q \geq 0$ for the same reason. For $q < 0$, we need only note that $L_\beta(q\mathbf{x}_i) = -q\boldsymbol{\beta}\cdot\mathbf{x}'_{(i)} = -q\boldsymbol{\beta}\cdot\mathbf{x}_{(i)}$ where $x'_{(k):i} = x_{(2N+2-k):i}$, $k = 1, \ldots, 2N+1$, since $\boldsymbol{\beta}$ is symmetric. ♦♦

Theorem 3: *The following two statements are equivalent:*

(1) $\{L_\alpha(x_i)\} = \{F_\alpha(x_i)\}$ *for any and every input sequence* $\{x_i\}$.

(2) $\alpha = q\mathbf{1}$ *for some real scalar* q.

Proof: That (2) implies (1) is obvious, since the data values are equally weighted on output regardless of their ordering.

To show that (1) implies (2), consider the sequence $\{x'_i\}$ having zero value everywhere except $x'_0 = -1$. Then $L_\alpha(\mathbf{x}'_0) = F_\alpha(\mathbf{x}'_0)$ implies that $\alpha_1 = \alpha_{N+1}$. Proceeding to the right we find that $L_\alpha(\mathbf{x}'_1) = F_\alpha(\mathbf{x}'_1)$ implies that $\alpha_1 = \alpha_N$. Continuing in this manner we find that $\alpha_1 = \ldots = \alpha_{N+1}$. Repeating this procedure by moving towards the left we find that $\alpha_1 = \alpha_{N+2} = \ldots = \alpha_{2N+1}$. It follows that $\{L_\alpha(x'_i)\} = \{F_\alpha(x'_i)\}$ only if $\boldsymbol{\alpha}$ is constant-valued. ♦♦

Thus, scalar multiples of the averaging filter are the only linear filters which yield the same output as OS filters defined with the same coefficients for *any* input. That this is so is not surprising, since averaging-type filters are the only type of filter which is both OS and linear.

B. Filters Defined with Symmetric Coefficients

In [10], it was noted that any symmetric OS and linear filter defined with identical coefficients yield the same output when applied to a monotonic sequence. We extend this in the following:

Theorem 4: *The following two statements are equivalent:*

(1) *The sequence* $\{x_i\}$ *is LOMO(2N+1).*

(2) $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ *for any and every symmetric (2N+1)-vector* β.

Proof: If $\{x_i\}$ is LOMO(2N+1), then every segment $\mathbf{x}_i$ of length 2N+1 is monotone. If $\mathbf{x}_i$ is nondecreasing, then $\mathbf{x}_{(i)} = \mathbf{x}_i$, so that $L_\beta(\mathbf{x}_i) = F_\beta(\mathbf{x}_i)$. If $\mathbf{x}_i$ is nonincreasing, then $\mathbf{x}_{(i)} = \mathbf{x}'_i = (x_{2N+1:i}, x_{2N:i}, \ldots, x_{1:i})^T$. Since $\boldsymbol{\beta}$ is symmetric, $L_\beta(\mathbf{x}_i) = \boldsymbol{\beta}^T\mathbf{x}'_i = \boldsymbol{\beta}^T\mathbf{x}_i = F_\beta(\mathbf{x}_i)$ so that (1) implies (2).

The proof that the converse holds is omitted here due to space limitations. ♦♦

While the class of signals for which $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ for all symmetric vectors $\boldsymbol{\beta}$ includes only LOMO(2N+1) sequences, we may achieve more specific results by applying certain restrictions on the filter coefficients. For the case of length three filters, we can arrive at a very simple result using Lemma 2 and Theorems 2 and 4.

Lemma 3: *Suppose that $\beta = (\beta_1, \beta_2, \beta_1)^T$ is an arbitrary length three symmetric vector such that $\beta_1 \neq \beta_2$. Then the following two statements are equivalent:*

(1) $\{x_i\}$ is LOMO(3).

(2) $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$.

Proof: That (1) implies (2) is evident from Theorem 4. To show that (2) implies (1), assume that β is as given and that $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$. Then from Lemma 2a we have that $\{L_{\beta'}(x_i)\} = \{F_{\beta'}(x_i)\}$ where $\beta' = (0, \beta_2-\beta_1, 0)^T$. Using Lemma 2d we also have that $\{L_{\beta''}(x_i)\} = \{F_{\beta''}(x_i)\}$ where $\beta'' = (0, 1, 0)^T$, which is the median/identity filter. Since any sequence contains a monotone segment of length 2 (= N+1), it follows that $\{x_i\}$ is LOMO(3) from Theorem 2. ♦♦

Lemma 3 is also a consequence of Theorem 5 which will be detailed later. It is important to note that this result cannot be generalized for symmetric nonconstant coefficient vectors of arbitrary length without some restriction on the input sequence and, as it turns out, on the coefficients. For example, consider the vector $\beta = (1, 0, 0, 0, 1)^T$. If $\{x_i\} = \{\ldots, 1, 0, 0, 1, 0, 1, 1, 0, \ldots\}$ where the pattern shown is repeated indefinitely in either direction, we note that $\{x_i\}$ is only LOMO(2) and yet $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$. However, we will find that we do have $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ if and only if the input is LOMO(4) for this particular filter if we require the input to contain at least one smooth (monotonic) region. The following Lemmas will be used to prove this more powerful result.

Lemma 4: *If $\{x_i\}$ is LOMO(n) and $k, m \geq n\text{-}1$, then for any nondecreasing (nonincreasing) and nonconstant segment $(x_p, \ldots, x_{p+k-1})$ and nonincreasing (nondecreasing) and nonconstant segment $(x_q, \ldots, x_{q+m-1})$, where $q = p+k$, there is a constant-valued segment of length at least n-1 containing both x_{p+k-1} and x_q.*

Proof: This Lemma simply states that any change in monotonic trend in a LOMO(n) sequence must include a constant-valued segment of length at least n-1. A simple proof is given in [7]. ♦♦

Lemma 5: *The following three statements are equivalent for $1 \leq k \leq N\text{-}1$:*

(1) $\{x_i\}$ is LOMO(N+k+2).

(2) $\{x_i\}$ is LOMO(N+k+1) and $\{x_{N-k+1:i}, x_{N+k+1:i}\} = \{x_{(N-k+1):i}, x_{(N+k+1):i}\}$ for every i.

(3) $\{x_i\}$ contains at least one monotonic region of length N+1, and for every $1 \leq j \leq k$ we have $\{x_{N-j+1:i}, x_{N+j+1:i}\} = \{x_{(N-j+1):i}, x_{(N+j+1):i}\}$ and $x_{N+1:i} = x_{(N+1):i}$ for every i.

Proof: We first show that (1) implies (2). If $\{x_i\}$ is LOMO(N+k+2), it is also LOMO(N+k+1). Consider the length 2N+1 segment $\mathbf{x}_i$. If $\mathbf{x}_i$ is nondecreasing, we have that $\{x_{N-k+1:i}, x_{N+k+1:i}\} = \{x_{(N-k+1):i}, x_{(N+k+1):i}\}$. Therefore, let us instead assume that the segment $(x_{1:i}, \ldots, x_{N+k+2:i})$ is nondecreasing and that $p = \min\{r: N+k+3 \leq r \leq 2N+1 \text{ and } x_{r-1:i} > x_{r:i}\}$. It follows from (1) that $x_{p-(N+k+1):i} = \ldots = x_{p-1:i}$, and that these elements are the largest contained in $\mathbf{x}_i$. Now, we can bound the index of the first element of this constant segment as $2 \leq p-(N+k+1) \leq N-k$, which shows that both $x_{N-k+1:i}$ and $x_{N+k+1:i}$ are elements of the segment. If we further note that at least N-k+1 elements of $\mathbf{x}_i$ are less than or equal to $x_{(N-k+1):i}$, and that at most N-k elements of $\mathbf{x}_i$ are less than the N+k+1 identical values, we have that $x_{(N-k+1):i}$ and $x_{(N+k+1):i}$ are both contained in the constant segment, and hence, are equal. In the above, we could have similarly assumed that $\mathbf{x}_i$ is a nonincreasing region with similar result.

To show that (2) implies (1), assume that $\{x_i\}$ is LOMO(N+k+1) but is not monotonic. Using Lemma 4, we may then assume that for some i we have $x_{1:i} \leq x_{2:i} = x_{3:i} = \ldots = x_{N+k+1:i} > x_{N+k+2:i}$. Then, we need only show that (2) implies that $x_{1:i} = x_{2:i}$. Since $1 \leq k \leq N\text{-}1$, also $2 \leq N\text{-}k\text{+}1 \leq N$, so that both $x_{N-k+1:i}$ and

$x_{N+k+1:i}$ are contained in the constant segment of length N+k+1. If $x_{1:i} < x_{2:i}$ there are exactly N-k+1 elements of x_i that are strictly less than $x_{N+k+1:i}$. Therefore $x_{N+k+1:i} \neq x_{(N-k+1):i}$ which contradicts (2). Therefore we must have $x_{1:i} = x_{2:i}$ and $\{x_i\}$ is LOMO(N+k+2). A similar result would be obtained by reversing the inequalities in our original assumption.

That (3) implies (2) is evident, since if $\{x_i\}$ contains a monotonic segment of length N+1 and $\{x_{N+1:i}\} = \{x_{(N+1):i}\}$, by Theorem 2 $\{x_i\}$ is LOMO(N+2). By iteration and using the fact that (1) <=> (2) we find that (3) implies (2).

We finish the proof by showing that (1) and (2) imply (3). By assuming (1) and (2), we note that (3) holds for j = k. But a signal that is LOMO(N+k+2) is also LOMO(N+k+1), so that (2) implies (3) for j = k-1. Again iterating we find that (3) holds for $1 \le j \le k$. ◆◆

Theorem 5 (Equivalence Theorem): *Suppose that the elements of the 2N+1 length symmetric vector* β *satisfy* $\beta_1 = \beta_2 = \ldots = \beta_{N-k}$ $(0 \le k \le N-1)$, *where one of the following hold:*

$$\text{(a) } \beta_1 < \min\{\beta_{N-k+1}, \ldots, \beta_{N+1}\}$$
$$\text{(b) } \beta_1 > \max\{\beta_{N-k+1}, \ldots, \beta_{N+1}\}.$$

Then, if $\{x_i\}$ *contains at least one monotonic (nonincreasing or nondecreasing) segment* $(x_p, \ldots, x_{p+N+k})$ *of length N+k+1, then* $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ *if and only if* $\{x_i\}$ *is LOMO(N+k+2).*

Proof: Using Lemma 2a we may assume that $\beta_1 = 0$. Let us assume that the segment $(x_p, \ldots, x_{p+N+k}) = (x_{1:i}, \ldots, x_{N+k+1:i})$ represents the first N+k+1 elements of x_i for some i, and without loss of generality that these elements are nondecreasing. If x_i is nondecreasing, then we have $L_\beta(x_i) = F_\beta(x_i)$. Let us therefore assume that for some $N+k+2 \le j \le 2N+1$ we have $x_{1:i} \le \ldots \le x_{N+k+1:i} \le \ldots \le x_{j-1:i} > x_{j:i}$. Then, $L_\beta(x_i) = F_\beta(x_i)$ if and only if

$$\sum_{n=N-k+1}^{N} \beta_n \left[x_{n:i} - x_{(n):i} + x_{2N+2-n:i} - x_{(2N+2-n):i} \right] + \beta_{N+1}\left[x_{N+1:i} - x_{(N+1):i} \right] = 0. \qquad (1)$$

Note that $x_{N+j+1:i} \ge x_{(N+j+1):i}$ for $-N \le j \le k$. If $\beta_{N-k+1}, \beta_{N-k+2}, \ldots, \beta_{N+1}$ are all positive (negative), then we simply have a sum of nonnegative (nonpositive) terms that sum to zero. Therefore each term must be zero, viz, $\{x_{N-j+1:i}, x_{N-j+1:i}\} = \{x_{(N-j+1):i}, x_{(N-j+1):i}\}$ for $1 \le j \le k$ and $x_{N+1:i} = x_{(N+1):i}$. If we were to begin again by assuming that $x_{1:i} \ge \ldots \ge x_{N+k+1:i} \ge \ldots \ge x_{j-1:i} < x_{j:i}$ we would achieve a similar result. Therefore, by Lemma 5 we have that $\{x_i\}$ is LOMO(N+k+2).

To prove the converse, we note that if $\{x_i\}$ is LOMO(N+k+2), then by Lemma 5 each term in (1) takes zero value, so that $L_\beta(x_i) = F_\beta(x_i)$. It should be noted that this implication does not require that $\beta_{N-k+1}, \beta_{N-k+2}, \ldots, \beta_{N+1}$ are of the same sign. ◆◆

The obvious question that arises is whether there are any filter vectors β other than those characterized in Theorem 5 for which we have $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ if and only if $\{x_i\}$ is LOMO. We believe that there are not. While we have not been able to show this for arbitrary N, we have been able to prove it to be so for $2N+1 \le 11$. Furthermore, we note again that characteristics (a) and (b) are not required for "sufficiency" in Theorem 5. We express this in the following conjecture:

Conjecture: *The only filters for which Theorem 5 is satisfied are those characterized in Theorem 5.*

Definition: If an OS/linear filter pair (L_β, F_β) have a coefficient vector β of length 2N+1 satisfying Theorem 5, then we say that they are *LOMO(N+k+2)-equivalent.*

Corollary 5.1: *Suppose that* (L_β, F_β) *are LOMO(N+k+2)-equivalent. If* β' *is of length 2M+1 (M > N), where* $\beta'_j = \beta_1$ *for* $j = 1, \ldots, M\text{-}N$ *and* $\beta'_j = \beta_{N-M+j}$ *for* $j = M\text{-}N+1, \ldots, M+1$*, then* $(L_{\beta'}, F_{\beta'})$ *are LOMO(M+k+2)-equivalent.*

Proof: Obvious, since if β satisfies the conditions of Theorem 5, then β' does also. ♦♦

Corollary 5.2: *Suppose that the sequence* $\{x_i\}$ *is LOMO(M), but not LOMO(M+1), and contains at least one segment of length N+k+1 which is monotone. If* (L_β, F_β) *are LOMO(N+k+2)-equivalent, then* $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ *if and only if* $M \geq N+k+2$.

Proof: It is clear that $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ if $M \geq N+k+2$, and equally clear that $\{L_\beta(x_i)\} \neq \{F_\beta(x_i)\}$ if $M < N+k+2$, from Theorem 5. ♦♦

Theorem 5 is an extension of Theorem 2, viz, median filters of length 2N+1 are LOMO(N+2)- equivalent to identity filters. Also, Lemma 3 may be regarded as a corollary to Theorem 5. There are several other well-known filters which are characterized by Theorem 5.

Definitions: A symmetric OS filter with coefficients $\beta_j = 0$ for $j = 1, \ldots, N\text{-}k$ and $\beta_j = 1/(2k+1)$ for $j = N\text{-}k+1, \ldots, N+1$ is known as a *k-inner mean filter* (also referred to as a trimmed mean filter). A symmetric OS filter with coefficients $\beta_j = 1/2k$ for $j = 1, \ldots, k$, and $\beta_j = 0$ for $j = k+1, \ldots, N+1$ is known as a *k-outer mean filter.*

Inner and outer mean filters have been noted to possess properties similar to those of both the linear averaging filter and the highly nonlinear median filter. Bednar and Watt [16] conjectured that repeated filterings of a given input sequence with a k-inner mean filter will eventually converge to the same result as obtained by repeatedly filtering the signal with a linear averaging filter of length 2k+1. The following corollary to Theorem 5 shows this result to be true for LOMO signals; here of course convergence is obtained in a single pass.

Corollary 5.3: *Linear averaging filters of duration 2k+1 and length 2N+1 k-inner mean filters are LOMO(N+k+2)-equivalent for* $N \geq k+1$.

Proof: A direct consequence of Theorem 5 and Corollary 5.1. ♦♦

A similar corollary holds for outer mean filters:

Corollary 5.4: *A k-outer mean filter of length 2N+1 and a linear filter defined with the same coefficients are LOMO(2N-k+2)-equivalent.*

Proof: Theorem 5. ♦♦

The class of filters considered in Theorem 5 includes a number of other linear filters of interest, including triangle-shaped filters, Gaussian-shaped filters of finite extent, and filters defined by differencing the outputs of averaging filters of different spans. Interestingly, it is filters of this type (among linear filters of finite span) which can be represented as morphological operations, within an additive constant. For linear filters of finite span, we have the following modified theorem of Maragos [11]:

Theorem 6: *Any linear filter of finite span whose coefficients are of constant sign can be represented exactly as a supremum of erosions followed by a single gain factor.*

Proof: See [11], p. 155. ♦♦

Thus, any linear filter satisfying Theorem 5 with $\beta_1 = 0$ can be represented as a morphological operation (of course, any OS filter can also be defined morphologically):

Corollary 6.1: *Suppose that the (L_β, F_β) are LOMO(N+k+2)-equivalent, and that $\beta_1 = 0$. Then F_β can be exactly represented by a supremum of erosions followed by a gain factor.*

Proof: By definition of LOMO(N+k+2) and from Theorem 6. ◆◆

In the following section, we will find that filters of this type enjoy certain smoothing properties as well (they are always low-pass, and they always preserve local monotonicty of sufficient order).

IV. SMOOTHNESS PRESERVING PROPERTIES OF LOMO-EQUIVALENT FILTERS

In the preceding we discussed several linear filters that can be shown to be LOMO-equivalent with their OS counterparts. We note that if a length (2N+1) linear filter F_β is LOMO(N+k+2)-equivalent to the OS filter L_β, and if $\beta_1 = 0$, then F_β is also a length 2k+1 filter with all positive or negative coefficients. In the following we show that such a filter is always of the "low-pass" type. Here we define $B(\omega)$ = frequency response of the linear filter with coefficient vector β.

Lemma 6: *Suppose that β is the length 2k+1 coefficient vector ($k > 0$) of a symmetric linear filter, where $\beta_i\beta_j > 0$ for $i \neq j$. Then $|B(0)| > |B(\omega)|$ for $\omega \in (0, \pi)$.*

Proof: For simplicity, assume that the impulse response $\{\beta_i\}$ of β is zero for $|i| > k$, and $\beta_i > 0$ for $|i| \leq k$. Then $|B(\omega)| = | \beta_0 + 2\{\beta_1\cos\omega + \beta_2\cos 2\omega + \ldots + \beta_k\cos k\omega\} | \leq \beta_0 + 2\{\beta_1|\cos\omega| + \beta_2|\cos 2\omega| + \ldots + \beta_k|\cos k\omega|\} < \beta_0 + 2\{\beta_1 + \beta_2 + \ldots + \beta_k\} = |B(0)|$, since $|\cos a\omega| < 1$ for $\omega \in (0, \pi)$, $a \neq 0$. ◆◆

Thus, linear filters having LOMO-equivalent OS counterparts are always low-pass, provided that the their outer coefficients are zero. Similarly, OS filters having LOMO-equivalent linear filters always act low-pass when applied to LOMO signals. In a related study [17], it was found that OS filtering of iid random sequences leads to sequences having maximum power at zero frequency, regardless of the filter coefficients. This suggests that OS filters have a smoothing tendency which is invariant to the coefficient vector, viz, that ordering the windowed data values imparts a high degree of correlation at the output. This is not the case for arbitrary linear filters, which suggests that OS filtered sequences are always smoother in some sense (probably monotonicity) than their linear counterparts, except when the input is LOMO. In the following we show that LOMO-equivalent filters with zero outer coefficients are also smoothness preserving in that they preserve local monotonicity.

Lemma 7 (Local Monotonicity Preserving Property): *Suppose that (L_β, F_β) are LOMO(N+k+2)-equivalent, and that $\beta_1 = 0$. Then if $\{x_i\}$ is LOMO(M), where $M \geq N+k+2$, then both $\{L_\beta(x_i)\}$ and $\{F_\beta(x_i)\}$ are LOMO(M).*

Proof: For any i, we may write

$$F_\beta(x_{i+1}) - F_\beta(x_i) = \sum_{j=N-k+1}^{N} \beta_j \left[x_{j:i+1} - x_{j:i} + x_{2N+2-j:i+1} - x_{2N+2-j:i} \right] + \beta_{N+1}\left[x_{N+1:i+1} - x_{N+1:i} \right]. \quad (2)$$

If $\{x_i\}$ is LOMO(M), where $M \geq N+k+2$, then for $k \leq N-1$ either $x_{j:i+1} - x_{j:i} \geq 0$ for $j = N-k+1, \ldots, N+k+1$ (when $x_{i+1} \geq x_i$) or $x_{j:i+1} - x_{j:i} \leq 0$ for $j = N-k+1, \ldots, N+k+1$ (when $x_{i+1} \leq x_i$). Since (L_β, F_β) are LOMO(N+k+2)-equivalent, and $\beta_1 = 0$, either (a) $\beta_{N-k+1}, \beta_{N-k+2}, \ldots, \beta_{N+1} > 0$ or (b) $\beta_{N-k+1}, \beta_{N-k+2}, \ldots, \beta_{N+1} < 0$. In case (a), if $x_{i+1} \geq x_i$ then all terms in (2) are nonnegative and $F_\beta(x_{i+1}) \geq F_\beta(x_i)$. If $x_{i+1} \leq x_i$ then $F_\beta(x_{i+1}) \leq F_\beta(x_i)$. Hence, F_β everywhere preserves monotonicity and $F_\beta(x_i)$ is LOMO(M). Similarly, in case (b), if $x_{i+1} \leq x_i$ then $F_\beta(x_{i+1}) \geq F_\beta(x_i)$, and if $x_{i+1} \geq x_i$ then $F_\beta(x_{i+1}) \leq F_\beta(x_i)$. Hence, F_β everywhere reverses monotonicity and $F_\beta(x_i)$ is LOMO(M). By Theorem 5 $\{L_\beta(x_i)\} = \{F_\beta(x_i)\}$ and $L_\beta(x_i)$ is LOMO(M) also. ♦♦

OS filters satisfying the conditions of Lemma 7 tend to be more robust than their linear counterparts since the extreme data values are "trimmed" away. This suggests that OS filtering might always lead to a "more LOMO" result than linear filtering, except when the input is LOMO. Thus, it would be of value to consider what we mean when we say that a sequence is "more LOMO" than another sequence; perhaps it is possible to show that the "closest" LOMO sequence to an arbitray sequence always exists and is unique. If so, it may be that OS filtering of the original sequence will always yield a result which is closer to the LOMO sequence than by linear filtering with the same coefficients. These topics and other related ones are currently under investigation.

REFERENCES

[1] J.W. Tukey, *Exploratory Data Analysis.* Reading, MA: Addison-Wesley, 1971, preliminary edition.

[2] L.R. Rabiner, M.R. Sambur, and C.E. Schmidt, Applications of a nonlinear smoothing algorithm to speech processing, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-23, pp. 552-557, Dec. 1975.

[3] B.R. Frieden, A new restoring algorithm for the preferential enhancement of edge gradients, *J. Opt. Soc. Amer.,* vol. 66, pp. 280-283, March 1976.

[4] T.S. Huang, G.J. Yang, and G.Y. Tang, A fast two-dimensional median filtering algorithm, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-27, pp. 13-18, Jan. 1979.

[5] E. Ataman, V.K. Aatre, and K.M. Wong, A fast method for real-time median filtering, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-28, pp. 415-420, Aug. 1980.

[6] N.C. Gallagher and G.L. Wise, A theoretical analysis of the properties of median filters, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-29, pp. 1136-1141, Dec. 1981.

[7] S.G. Tyan, Median filtering: deterministic properties, in *Two-Dimensional Signal Processing: Transforms and Median Filters,* (T.S. Huang, Ed.). New York: Springer-Verlag, 1981.

[8] T.A. Nodes and N.C. Gallagher, Jr., Median filters: some modifications and their properties, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-30, pp. 739-746, Oct. 1982.

[9] A.C. Bovik, T.S. Huang, and D.C. Munson, Jr., A generalization of median filtering using linear combinations of order statistics, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-31, pp. 1342-1350, Dec. 1983.

[10] Y.H. Lee and S.A. Kassam, Generalized median filtering and related nonlinear filtering techniques, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-33, pp. 672-683, June 1985.

[11] P.A. Maragos, A unified theory of translation-invariant systems with applications to morphological analysis and coding of images, Georgia Institute of Technology School of Electrical Engineering Tech. Rept. DSPL-85-1, July 1985.

[12] P.D. Wendt, E.J. Coyle, and N.C. Gallagher, Stack filters, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-34, pp. 898-911, Aug. 1986.

[13] E.J. Coyle, On the optimality of rank order operations under the absolute error criterion, in *Proc. Twentieth Annual Conf. Info. Sci. Syst.,* Princeton, NJ, pp. 705-710, March 1986.

[14] A.C. Bovik and L. Naaman, Least-squares signal estimation using order statistic filters, in *Proc. Twentieth Annual Conf. Info. Sci. Syst.,* Princeton, NJ, pp. 735-739, March 1986.

[15] F. Palmieri and C.G. Boncelet, Jr., Optimal mse linear combination of order statistics for restoration of Markov processes, in *Proc. Twentieth Annual Conf. Info. Sci. Syst.,* Princeton, NJ, pp. 729-734, March 1986.

[16] J.B. Bednar and T.L. Watt, Alpha-trimmed means and their relationship to median filters, *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-32, pp. 145-153, Feb. 1984.

[17] A. Restrepo and A.C. Bovik, Spectral analysis of order statistic filters, in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing,* Tokyo, Japan, April 7-11, 1986.

SUCCESSIVE PARTITIONING AND SELECTION FILTERS*

YONG HOON LEE, YONGJOO LEE and ADLY T. FAM
Department of Electrical and Computer Engineering
State University of New York at Buffalo
Buffalo, New York 14260
Tel. (716) 636-2406

ABSTRACT

Two nonlinear techniques enhancing blurred edges in the abscence of knowledge about the blur are presented. These methods can be thought of as adaptive order statistic filters that select a sample from each window moving across the input sequence. The sample selection is performed by successively partitioning the data inside each window based on averaging operation. It is shown that these filters can enhance blurred edges with various shapes while suppressing impulsive and some nonimpulsive additive white noise components. These filters are applied to actual blurred images, and their performances are compared with that of a conventional edge enhancing technique employing the Laplacian operator. The result shows that the proposed methods outperform the conventional one when the input image is noisy and/or severly blurred.

I. INTRODUCTION

Median filtering has been recognized as a useful digital signal processing technique in many applications which include image, speech and vibration signal analysis [1],[2]. The median filtered sequence is obtained by replacing the input value at each point with the median value of the data inside the window centered at the point. One useful property of the median filter is its ability to smooth the input signal while preserving signal edges. In particular, impulses can be effectively filtered out by median filtering. Recently, some generalizations of median filtering and related filtering techniques have been proposed [3]-[6]. These filters have characteristics similar to those of median filtering, and are often referred to as order statistic filters.

In many applications, edge-preserving filtering is important. In some cases, however, it is necessary to enhance gradients or slopes of edges rather than to simply preserve them. For example, in image processing, image degradation generally involves blurring. Blurred images can be enhanced by enhancing edge gradients. Median filters can preserve edges but cannot enhance them.

In general, blurred edges can be enhanced by linear deconvolution if the knowledge about the blur is available. Friden [7] proposed an edge gradient enhancing technique that uses linear deconvolution in combination with median filtering, where median filtering is employed to eliminate the Gibbs

---

*This work was supported by the National Science Foundation under Grant DCI-8611859.

oscillations in linear deconvolution. In cases where the knowledge about the blur is not available, some linear technique based on differentiation or linear high frequency emphasis filtering have been proposed [2]. While these techniques can enhance blurred edges, they are usually sensitive to noise.

Recently a nonlinear method that can suppress noise while enhancing edge gradients has been proposed [8]. This method is called the comparison and selection (CS) filter, and can be thought of as an adaptive order statistic filter. Like the other order statistic filters, the CS filter is particularly effective in suppressing impulsive noise components. CS filtering is a useful edge enhancing technique in the absence of knowledge about the blur.

In this paper, motivated by CS filtering, we introduce other adaptive order statistic filters for the enhancement of edges. The proposed methods have some advantages over the CS filter. Before describing these methods, a short review of previous work in CS filtering follows.

The Comparison and Selection Filter [8].

The output of the CS filter is determined by comparing the sample mean with the sample median inside each window. If {X(k)} and {Y(k)} are the input and the output sequences of the CS filter with window size 2N+1, then

$$Y(k) = \begin{cases} X_k^{(N+1-J)} & \text{if} \quad \mu(k) \geq M(k) \\ X_k^{(N+1+J)}, & \text{otherwise} \end{cases}$$

where J is an integer satisfying $1 \leq J \leq N$, $X_k^{(i)}$ is the i-th smallest sample inside the window at k, and $\mu(k)$ and M(k) are the sample mean and the sample median, respectively.

It is shown theoretically that CS filters can enhance blurred edges of various shapes including noisy blurred ones. Experimental results show that CS filtering is useful in enhancing noisy blurred images. While CS filters are insensitive to noise, a tradeoff between edge enhancing and noise suppression properties has been observed, that is, the CS filter with a large value of J (e.g., J=N) is better in enhancing edges but worse with regard to noise suppression, compared with the CS filter with a small value of J. The parameter J should be chosen to get a good compromise between edge enhancement and noise suppression.

This paper describes two methods that can overcome the difficulty encountered in designing the CS filter. Both of these methods involve successive partitioning of the data inside each window based on averaging operation, and are referred to as successive averaging and selection (SAS) and successive midranging and selection (SMS) filters. In Sections II and III we introduce SAS and SMS filters, respectively. We shall see that the only parameter to be determined for these methods is the window size as in the case of median filtering, and show that a variety of blurred edges can be enhanced by these methods. In addition, it is shown that SAS and SMS filters can outperform CS filters in enhancing edges that are severely blurred and corrupted by impulsive noise. Section IV presents some simulation and experimental results. Throughout this paper we assume that the input and output sequences are one dimensional and two sided infinite unless otherwise specified. We use odd window sizes which we write as 2N+1 for a positive integer N. The sample mean and median values in the window centered at point k will be denoted by $\mu(k)$ and $M(k)$, respectively.

II. THE SUCCESSIVE AVERAGING AND SELECTION FILTER

Consider the set of samples inside a window. We partition the samples into two subsets where one subset is composed of the samples greater than the sample mean and the other is composed of the rest of the samples. It is significant to observe that the output value of a CS filter is chosen from the subset that contains the sample median, and the subset has more samples than the other. This observation suggests the following SAS filtering operation for edge enhancement, where we initially consider all 2N+1 samples within the window.

Step 1: If all of the data in the set are the same in value, take that value, otherwise go to Step 2.

Step 2: Evaluate the mean of samples in the set.

Step 3: Partition the samples in the set into two subsets, where one subset consists of samples greater than the mean and the other consists of the rest.

Step 4: Retain the subset having more samples and discard the other. If the number of samples in the retained subset is even, then discard an additional sample from it whose value is the closest or equal to the mean.

Step 5: If Step 4 results in only one retained sample, then select it as the output value. Otherwise, return to Step 1 with the set of retained samples.

The SAS filter selects one of the ordered value inside the window depending on the data. The i-th smallest sample is selected as the output only if i is odd. Thus the sample median which is the (N+1)-th smallest sample can be the output value if N is even.

SAS filters can outperform CS filters. We shall show later in this section that the SAS filter enhances edges like the CS filter with a large value of J, specifically $J = N$. On the other hand, the SAS filter suppresses impulsive noise components like the CS filter with $J = 0$ that is the median filter. The impulsive noise suppression characteristic of the SAS filter can be conjectured from the following property.

Property 1 : If there exist N+1 samples having the same value, say x, in the window of size 2N+1, the output of SAS filtering is x regardless of the other N sample values.

The proof is omitted. Property 1 also holds for median filtering, but not for CS filtering with $J > 1$. The impulsive noise suppression property of the CS filter depends on the parameter J. If J is large, impulses may not be well suppressed by CS filtering [8]. In addition, the edge enhancing behavior of CS filtering can be degraded if impulses exist in the neigborhood of blurred edges. For example, the ramp edge in Fig. 1(a) become the ideal step edge in Fig.1(b) after CS filtering with $2N+1=7$ and $J=2$. As depicted in Fig. 1(c) and (d), however, when an impulse exists in the ramp edge the CS filter suppresses the impulse but cannot enhance the edge. Due to the successive partitioning operation, the edge enhancing behaviors of SAS filters are not impaired by the existance of impulses in the neighborhood of blurred edges. For example, again referring to Fig. 1, the noisy blurred edge in Fig. 1(c) becomes the ideal step edge in Fig. 1(b) after SAS filtering with $2N+1 = 7$. Therefore, in general, the behavior of the SAS filter is preferred to that of the CS filter when the edges to be enhanced are severely blurred and corrupted by impulses. This will be demonstrated through experimental results in Section IV.

In the following, we show theoretically that SAS filters can enhance a variety of edges. The proofs for the edge enhancing properties stated below are omitted. Lemmas 1, 2 and Property 2 below help us to establish the other property that considers the enhancement of ramp and convex/concave edges [8] (also see Definition 2). Lemmas 1 and 2 describe the results which have been proved in [8]. We first define convex and concave sequences.

Definition 1: A sequence $\{X(i)\}$ is convex(concave) if $d(i) \geq d(i-1)$ ($d(i) < d(i-1)$) for each i where $d(i) = X(i)-X(i-1)$.

Here a ramp sequence ($d(i) = d(i-1)$ for each i) is reckoned as a convex one because SAS filters inherently treat ramp sequences as convex sequences.

Lemma 1 [8] : If the sequence is convex(concave), then $\mu(i) \geq M(i)$ ($\mu(i) < M(i)$) for all i.

From Lemma 1, the following property can be proved.

Property 2: If the input $\{X(i)\}$ of the SAS filter is strictly increasing(decreasing) and convex(concave) then the output $\{Y(i)\}$ is given by

$$Y(i) = X(i-N) \quad \text{for all } i.$$

Similarly, if $\{X(i)\}$ is strictly increasing(decreasing) and concave(convex) then,

$$Y(i) = X(i+N)$$

In many cases an increasing blurred edge is composed of increasing convex part followed by a concave one. Such edge is called the convex/concave edge (see Definition 2). The ramp edge is a special case of the convex/concave edge. It is conjectured from Property 2 that the convex part is delayed and the concave part is advanced by SAS filtering of a convex/concave edge. This results in shrinking of the convex/concave edge, thus enhancing it. CS filters also shift convex or concave sequences, but the amount of shifting is J instead of N [8].

The analysis of edge enhancing behaviors of SAS filters requires a blurred edge model. The blurred edge model used is the sequence $\{X(i)\}$ which is defined by

$$X(i) = \begin{cases} 0 , & i \leq 0 \\ f(i) , & 0 < i \leq D \\ H , & D < i \end{cases} \qquad (1)$$

where $\{f(i)\}$, $-\infty < i < \infty$, is a sequence that determines the blurred edge shape. The positive constants H and D parameterize the degree of blurring, which are the height and the duration of the edge, respectively. Here D is an integer. A convex/concave edge is defined as follows :

Definition 2: $\{X(i)\}$ in (1) is a convex/concave edge if $d(i) \geq d(i-1)$, $1 \leq i \leq T$, and $d(i) < d(i-1)$, $T < i \leq D+1$, where T is an integer satisfying $1 \leq T \leq D+1$.

Note that, at point T, $d(T) \geq d(T-1)$ and $d(T) > d(T+1)$. The point T is the transition point between convex and concave regions. The blurred edge model in (1) has been used to analyze the edge enhancing behaviors of CS filters [8] and will also be used for SAS filters. This enables us to theoretically compare the performances of CS and SAS filters in enhancing edges.

Lemma 2 [8]: If $\{X(i)\}$ in (1) is a convex/concave edge, then there exists an integer n for which $\mu(i) \geq M(i)$ if $i \leq n$, and $\mu(i) < M(i)$ if $i > n$ for any i, $-N+1 \leq i \leq D+N$, where $T-N \leq n \leq T+N$, and T is the transition point between convex and concave regions, and $2N + 1$ is the window size.

Now we are ready to characterize the behavior of the SAS filter in enhancing convex/concave edges.

Property 3: Any convex/concave edge with duration D becomes an ideal step edge after at most $\lfloor D/N \rfloor + 1$ passages through the SAS filter with the window size 2N+1, where $\lfloor x \rfloor$ represents the integer part of x. The transition from zero to H of the resulting step edges occurs between T-N and T+N.

Property 3 shows that SAS filters can enhance convex/concave edges with various shapes. In [8], it is shown that the CS filter can also enhance convex/concave edges. Specifically, it requires $\lfloor D/J \rfloor + 1$ iterations to produce an ideal step edge. Since the SAS filter needs $\lfloor D/N \rfloor + 1$ iterations to convert a convex/concave edge to an ideal step edge, the SAS filter performs like the CS filter with J = N in enhancing convex/concave edges. It can be shown that SAS filters can enhance blurred edges other than convex/concave ones. In the following property, we show that a SAS filter can enhance an edge with an arbitrary shape if its window size is sufficiently large.

Property 4 : Any blurred edge represented by (1) becomes an ideal step edge after at most two passages through the SAS filter with $N \geq 2D - 1$.

Properties 3 and 4 generally suggest the use of SAS filters with large window sizes. However, the window size of the SAS filter is limited by the distances between neighboring edges in the input sequence.

III. THE SUCCESSIVE MIDRANGING AND SELECTION FILTER

In SAS filtering, the data inside the window is successively partitioned based on simple averaging operation in which the data to be averaged are weighted equally. In this section, we describe a modification of the SAS filter which is called the successive midranging and selection filter (SMS) where a weighted average is employed in liu of the simple average. Specifically, the mean of the sample minimum and the sample maximum that is called the midrange [9] is used. The midrange is a weighted average of the ordered data. We shall show that the SMS filter performs like the SAS filter in enhancing edges and in suppressing noise, while the former is easier to implement than the latter. The midrange of the data within the window at point k is denoted by A(k).

The SMS filtering procedure is identical to that of SAS filtering except in Step 2. In Step 2 of SMS filtering the midrange that is the average of the minimum and the maximum values in the set is calculated instead of averaging all of the values in the set. The partitioning procedure in Steps 3 and 4 is performed based on the midrange.

The properties of SMS filters are similar to those of SAS filters. In fact some of the properties of SMS filters are identical to the properties of SAS filters. Next we describe the properties of SMS filters corresponding to those of SAS filters discussed in Section II.

The SMS filter can also suppress impulses like the median filter. This characteristic can be conjectured from the property below that corresponds to Property 1 in the previous section.

Property 5: If there exist N+1 samples having the same value, say x, in the window of size 2N+1, the output of SMS filtering is x regardless of the other sample values.

It can be shown that SMS filters also enhance convex/concave edges. We state the result as follows:

Lemma 3: If the sequence is convex(concave), then $A(i) \geq$

$M(i)$ ($A(i) < M(i)$) for all i.

Property 6: If the input $\{X(i)\}$ of the SMS filter is strictly increasing(decreasing) and convex(concave) then the output $\{Y(i)\}$ is given by

$$Y(i) = X(i-N) \quad \text{for all } i.$$

Similarly, if $\{X(i)\}$ is strictly increasing(decreasing) and concave(convex) then,

$$Y(i) = X(i+N) \quad \text{for all } i.$$

Lemma 4: If $\{X(i)\}$ in (1) is a convex/concave edge, then there exists an integer n for which $A(i) \geq M(i)$ if $i \leq n$, and $A(i) < M(i)$ if $i > n$ for any i, $-N+1 \leq i \leq D+N$, where $T-N \leq n \leq T+N$, and T is the transition point between convex and concave regions.

Now we are ready to show the convex/concave edge enhancing property of the SMS filter.

Property 7: Any convex/concave edge with duration D becomes an ideal step edge after at most $\lfloor D/N \rfloor + 1$ passages through the SMS filter with the window size $2N + 1$. The transition from zero to H of the resulting step edge occurs between $T - N$ and $T + N$. Here T denotes the transition point between the convex and concave regions.

The following property corresponds to Property 4 of SAS filtering.

Property 8: After at most two passages through the SMS filter with $N \geq 2D - 1$, any blurred edge represented by (1) becomes an ideal step edge.

So far we have observed that all of the properties of SAS filters described in Section II also hold for SMS filters. Now we present some edge enhancing properties of SMS filters for which SAS counterparts are not exist. First let us describe a modification of Property 8.

Property 9: After one passage through the SMAS filter with $N \geq 2D - 1$, any blurred edge represented by (1) becomes an ideal step edge if $0 < f(i) < H$ for $1 \leq i \leq D$.

In Property 8 and 9, the size of the window enhancing edges with arbitrary shapes is rather large. If the values in the edge is limited between 0 and H as in Property 9, it can be shown that the SMS filter with a smaller window size can enhance the blurred edge regardless of its shape.

Property 10: After at most $D - (N + 1)/2 + 1$ uses of the SMS filter with $D \leq N < 2D - 1$, any blurred edge represented by (1) becomes an ideal step edge as long as $0 < f(i) < H$ for $1 \leq i \leq D$.

IV. EXPERIMENTAL RESULTS

In this section, the filters we have discussed so far are applied to enhance edges of images. The filtering operations considered in this section are separable filtering in which each row of the input image is passed through a 1D filter, and then each column of the resulting image is passed through the same 1D filter. A window size 2N + 1 = 7 was used for these 1D filters.

In image processing, the Laplacian operator is often used to enhance edges [2], that is given by

$$\nabla^2 X(i,j) = [X(i+1,j) + X(i-1,j) + X(i,j+1) + X(i,j-1)] - 4X(i,j)$$

where X is the input image and $\nabla^2 X$ denotes the Laplacian operator. The edge enhanced image is obtained by subtracting $\nabla^2 X(i,j)$ from $X(i,j)$ for all i, j. In our experiments, this edge enhancing technique, $X - \nabla^2 X$, is also applied for comparison purposes.

Fig. 2(a) shows the original image of a plier. The original image in Fig. 2(a) is blurred by passing it twice through the mean filter with rectangular window of size 5x5. Here the mean filter replaces the center value of each window with the average of the values inside the window. The blurred image is corrupted by impulses with probability of occurance 0.1. The noisy blurred image is shown in Fig. 2(b). The noisy blurred image is processed through the median filter followed by $X - \nabla^2 X$, the CS filter with $J = 1$, the CS filter with $J = 3$, the SAS filter and the SMS filter. Here the median filter is used prior to $X - \nabla^2 X$ in order to suppress impulses. The results are shown in Fig. 2(c)-(g). In this case, the SAS and SMS filters outperform the median filter followed by $X - \nabla^2 X$ and the CS filters. The SAS and SMS filters are more effective in enhancing edges compared with $X - \nabla^2 X$ and the CS filter with $J = 1$. The edges were not properly enhanced by the CS filter with $J = 3$ because of the impulses in the neighborhood of the edges. In general, when edges are severely blurred and corrupted by impulses the performances of SAS and SMS filters are preferred to those of CS filters.

REFERENCES

1. L.R. Rabiner, M.R. Sambur, and C.E. Schmidt, "Applications of a nonlinear smoothing algorithm to speech processing," IEEE Trans. Acoust., Speech, and Signal Proc., vol. ASSP-23, pp. 552-557, Dec. 1975.

2. A. Rosenfeld and A.C. Kak, Digital Picture Processing, vol.1, 2nd ed., Academic press, 1982.

3. A.C. Bovik, T.S. Huang, and D.C. Munson, Jr., "A generalization of median filtering using linear combinations of order statistics," IEEE Trans., Acoust., Speech, and

Signal Proc., vol. ASSP-31, pp. 1342-1350, Dec. 1983.

4. J.B. Bedner and T.L. Watt, "Alpha-trimmed means and their relationship to median filters,"IEEE Trans. Acoust., Speech, and Signal Proc., vol. ASSP-32, pp. 145-153, Feb. 1984.

5. Y.H. Lee and S.A. Kassam, "Generalized median filtering and related nonlinear filtering techniques,"IEEE Trans. Acoust., Speech, and Signal Proc., vol. ASSP-33, pp. 672-683, June 1985.

6. A.R. Butz, "A class of rank order smoothers," IEEE Trans., Acoust., Speech, and Signal Proc., vol.ASSP-34, pp.157-165, Feb. 1986.

7. B.R. Friden, "A new restoring algorithm for the preferential enhancement of edge gradients," J. Opt. Soc. Am., vol. 66, pp. 280-283, Mar. 1976.

8. Y.H. Lee and A.T. Fam, "An edge gradient enhancing adaptive order statistic filter," Accepted for publication pending minor revision in IEEE Trans. Acoust., Speech, Signal Proc. Portions of this paper were published in Proc. 20th Annual Conf. on Informance Sciences and Systems, pp. 717-722, March 1986.

9. H.A. David, Order Statistics, 2nd ed., Wiley, 1981.

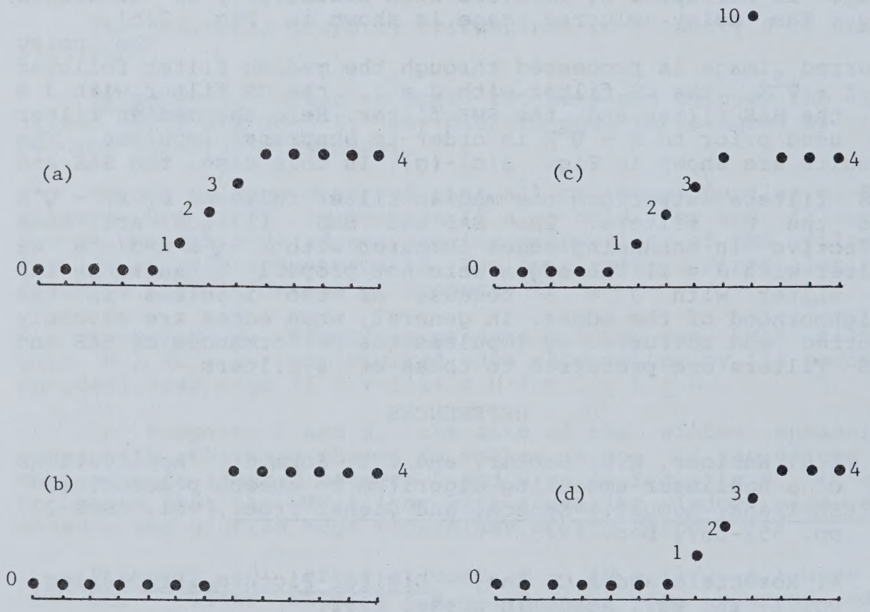


Fig. 1. (a) A ramp edge. (b) CS filtered sequence of the ramp in (a) or SAS filtered sequence of the one in (c) (2N+1=7). (c) A ramp edge corrupted by an impulse. (d) Result of CS filtering of (c).

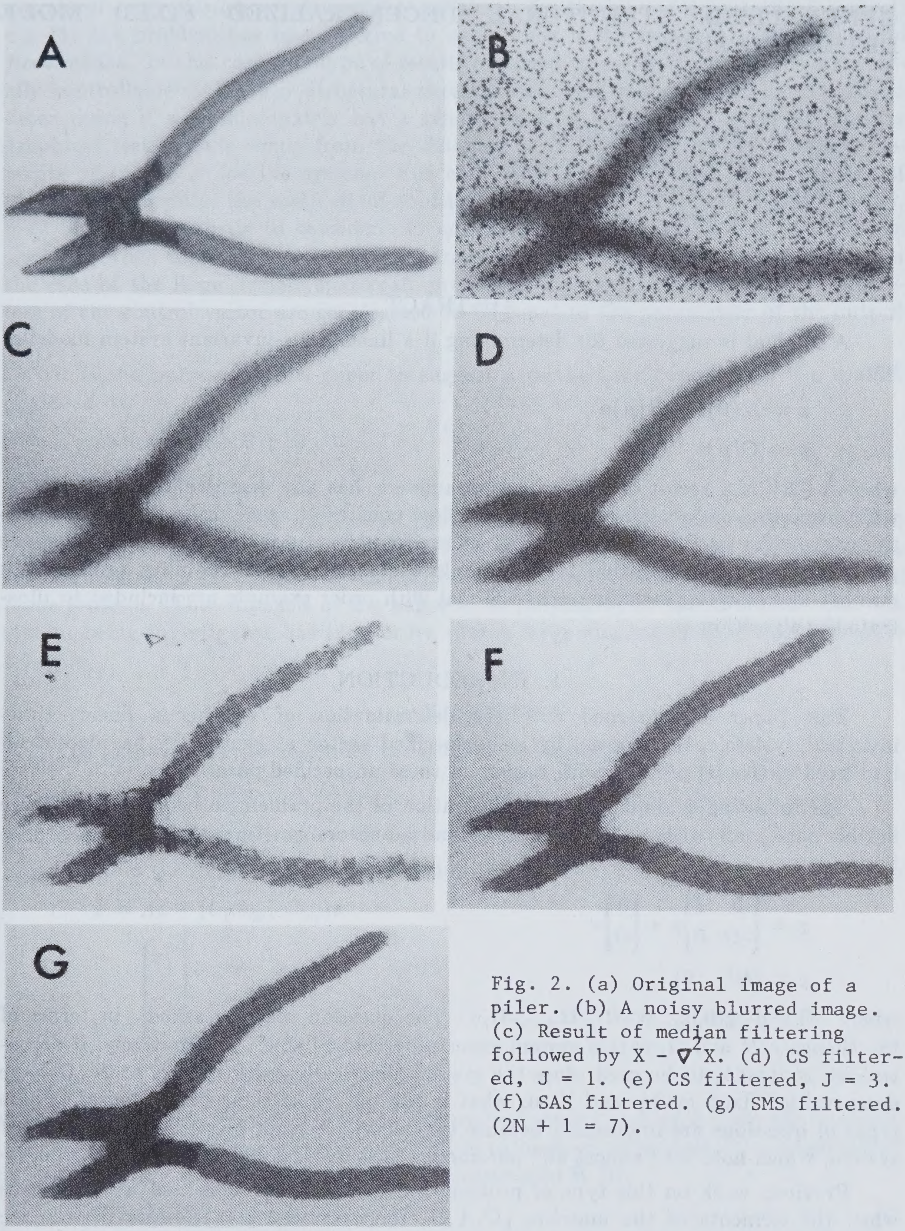


Fig. 2. (a) Original image of a piler. (b) A noisy blurred image. (c) Result of median filtering followed by $X - \nabla^2 X$. (d) CS filtered, $J = 1$. (e) CS filtered, $J = 3$. (f) SAS filtered. (g) SMS filtered. $(2N + 1 = 7)$.

GENERALIZED STRUCTURAL DECENTRALIZED FIXED MODE DETERMINATION

E.J. DAVISON, W. GESING
Department of Electrical Engineering
University of Toronto
Toronto, Ontario, Canada M5S 1A4

ABSTRACT

A method is suggested for determining if a linear time-invariant system modelled by:

$$\dot{x} = A(p)x + B(p)u$$
$$y = C(p)x$$

where $p \in R^{\gamma}$ is a vector of unspecified parameters, has any decentralized fixed modes, with respect to an arbitrary information flow constraint, generically with respect to the parameters $p \in R^{\gamma}$. This called the generalized structural decentralized fixed mode problem, and is an extension to the standard structural controllability problem. A number of examples, including a non-trivial 20th order example are included to illustrate the algorithm.

1. INTRODUCTION

This paper is concerned with the determination of whether a linear, time-invariant system, parametrized by an unspecified vector p, generically has decentralized fixed modes [1] present, with respect to these unspecified parameters.

The following example gives an illustration of the problem to be studied: a large flexible spacecraft with co-located sensors and actuators can be modelled by the equations:

$$\dot{x} = \begin{pmatrix} 0 & I \\ -\Omega & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ B \end{pmatrix} u$$
$$y = (B' \quad 0)$$

where $-\Omega \triangleq \text{diag}(0,0,\ldots,0;-\Omega_1,-\Omega_2,\ldots,-\Omega_t)$. The question may be asked: in terms of the elements B and Ω, is this system generically controllable? Alternatively, if decentralized control is to be used, does the system generically, with respect to B, Ω have decentralized fixed modes and if so, what is the nature of these fixed modes? These types of questions are important, because they specify certain properties for a class of system, which hold for "almost all" parameters of such class.

Previous work on this type of problem has generally be concerned with the case when the elements of the matrices (C,A,B) for a system $\dot{x}=Ax+Bu$, $y=Cx$ are either zero or unspecified, with no constraint imposed on the unspecified elements. For the case of centralized control, e.g. see [1]-[5], the problem has been called the

---

This work has been supported by the Natural Sciences and Engineering Research Council of Canada under grant no. A4396.

structural controllability or observability problem - for the case of decentralized control, e.g. [6], the problem has been referred to as deciding if the system has any *structural fixed modes.* In this case, the type of results obtained to decide if a system is structurally controllable or has any structural decentralized fixed modes, typically depend on determining if a certain matrix has a generic rank of specified order, or on various graphical tests which result from the detailed interconnections of the non-zero elements of C, A, B for the system. For high order systems with a large number of inputs and outputs, the methods of solution suggested in, e.g. [1]-[6], unfortunately, may become unrealistic to consider. In addition, they typically do not consider the situation when some of the unspecified parameters of (C,A,B) are constrained, e.g. in the case of the large flexible spacecraft previously discussed, the unspecified parameters of the control vector are constrained to be equal to the parameters of the output vector.

It is the purpose of this paper to suggest a method for determining if a system modelled by:

$$\dot{x} = A(p)x + B(p)u$$

$$y = C(p)x$$

where $p \in R^{\gamma}$ is a vector of unspecified parameters, has any decentralized fixed modes, with respect to an arbitrary information flow constraint, generically, with respect to the parameters $p \in R^{\gamma}$. This is called the *generalized structural decentralized fixed mode problem.* The emphasis of the paper is, in particular, on the situation when the system being investigated has high order with a large number of inputs and outputs present.

2. DEVELOPMENT

The following section describes the problem to be solved.

2.1 *Controller Information Flow Constraint*

Consider the linear time-invariant plant:

$$\dot{x} = Ax + (b_1, b_2, \ldots, b_m)u$$

$$y = \begin{pmatrix} c_1' \\ c_2' \\ \vdots \\ c_\nu' \end{pmatrix} \qquad (1)$$

where $x \in R^n$ is the state, $u \in R^m$ are the control inputs and $y \in R^r$ are the outputs. Assume that a controller $u = K(s)y$ which is to be used to control (1) is constrained by the following *controller information flow constraint* **K** [1]:

$$\mathbf{K} = \{(i_1, j_1), (i_2, j_2), \ldots, (i_s, j_s)\} \qquad (2)$$

where $i_k \in [1,2,\ldots,m]$, $j_k \in [1,2,\ldots,r]$, $k = 1,2,\ldots,s$, i.e. all elements of $K(s)$, except for the (i_k, j_k) elements corresponding to the i_kth row and j_kth column of $K(s)$, $k = 1,2,\ldots,s$ are constrained to be zero.

In this case, the *decentralized fixed modes of (1) with respect to* **K** [1] are given by all $\lambda \in \mathrm{sp}(A)$ which have the property that

$$\det\left\{A-\lambda I+\sum_{k=1}^{s} \mu_k b_{i_k} c'_{j_k}\right\}=0, \quad \forall \mu_k \in R, \; k=1,2,\ldots,s$$

and a simple algorithm is given in [10], [12] for determining such decentralized fixed modes.

Definition: Given a system (1), denoted by (C,A,B) and an information flow constraint **K** given by (2), the following notation shall be used to denote (3):

$$\det\{A-\lambda I+BKC\}=0, \quad \forall K \in \mathbf{K}$$

Remark 1: It is to be noted that the controller information flow constraint **K** is completely general, i.e. there always exists $s>0$ and a pairing (i_k, j_k), $k=1,2,\ldots,s$ so as to include all possible types of controller constraints, e.g. centralized control, etc.

The following result shows that decentralized fixed modes play an essential role in determining whether the system (1) can be stabilized by a controller which is subject to the information flow constraint **K**.

Theorem 1 [1]: There exists a controller to stabilize the system (1), subject to the information flow constraint **K**, if and only if the system (1) has no unstable decentralized fixed modes with respect to **K**.

It is therefore important to know under what conditions a system has decentralized fixed modes present. In particular, it is of interest to know under what conditions, a system generically has no decentralized fixed modes present with respect to a set of parameters.

2.2 *Unspecified Systems*

Assume that the matrix $M(p) \in R^{s \times t}$ is parametrized by the real γ dimensional vector p, where the ith, jth element of $M(p)$ is given by:

$$m_{ij}(p)=\frac{n_{ij}(p)}{d_{ij}(p)}, \quad d_{ij}(p) \neq 0 \tag{4}$$

where $n_{ij}(p)$ and $d_{ij}(p)$ are polynomials in the elements of p; then $M(p)$ is said to be an *unspecified matrix*, and $p \in R^{\gamma}$ are said to be the *unspecified elements* of M.

Consider now a system parametrized by a vector $p \in R^{\gamma}$

$$\begin{aligned} \dot{x} &= A(p)x + B(p)u \\ y &= C(p)x \end{aligned} \tag{5}$$

where $x \in R^n$ is the state, $u \in R^m$ is the input and $y \in R^r$ is the output, and where $\{C(p),A(p),B(p)\}$ are unspecified matrices, and $p \in R^{\gamma}$ are the unspecified elements of $\{C(p),A(p),B(p)\}$. Then (5) is said to be an *unspecified system.*

Consider now the unspecified system (5). Then if $\{C(p),A(p),B(p)\}$ has no decentralized fixed modes with respect to **K** for almost all [7] parameters $p \in \Omega$, the system $\{C(p),A(p),B(p)\}$ is said to have no *structural decentralized fixed modes with respect to* **K**.

The following gives some simple illustrations of unspecified systems:

Example 1': A dc motor can be modelled by the following equations:

$$\dot{x} = \begin{pmatrix} -\dfrac{B}{J} & \dfrac{K_T}{J} \\ -\dfrac{K_e}{L} & -\dfrac{R}{L} \end{pmatrix} x + \begin{pmatrix} 0 \\ \dfrac{1}{L} \end{pmatrix} u \,; \quad x \in R^2, \; u \in R, \; y \in R$$

$$Y = (1 \quad 0) x \tag{6}$$

where $J > 0$, $L > 0$, $B > 0$, $K_e > 0$, $K_T > 0$, $R > 0$. In this case (6) corresponds to the unspecified system (5) with $p \in R^6$ where:

$$A(p) \triangleq \begin{pmatrix} -\dfrac{p_2}{p_1} & \dfrac{p_3}{p_1} \\ -\dfrac{p_4}{p_6} & -\dfrac{p_5}{p_6} \end{pmatrix}, \quad B(p) \triangleq \begin{pmatrix} 0 \\ \dfrac{1}{p_6} \end{pmatrix}, \quad C(p) \triangleq (1 \quad 0)$$

Example 2': Consider the unspecified system (5) with $p \in R^2$ and

$$C(p) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad A(p) = \begin{pmatrix} 1 & -1+p_1 \\ 2 & -2+p_1 \end{pmatrix}, \quad B(p) = \begin{pmatrix} p_2 \\ p_2 \end{pmatrix}$$

In this case, if a centralized controller is used, i.e. $\mathbf{K} = \{(1,1),(1,2)\}$, it can be easily verified that this system has a structural fixed mode with respect to $\mathbf{K}$ present, i.e. is uncontrollable $\forall p \in R^2$.

Example 3': Consider the unspecified system (5) with $p \in R^8$ and

$$C(p) = \begin{pmatrix} p_7 & 0 & 0 & 0 \\ 0 & 0 & p_8 & 0 \end{pmatrix}, \quad A(p) = \begin{pmatrix} 0 & p_1 & 0 & p_2 \\ 0 & 0 & 0 & 0 \\ 0 & p_3 & 0 & p_4 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad B(p) = \begin{pmatrix} 0 & 0 \\ p_5 & 0 \\ 0 & 0 \\ 0 & p_6 \end{pmatrix}$$

In this case if the information constraint $\mathbf{K} = \{(1,1),(1,2),(2,1),(2,2)\}$ is used, i.e. centralized control, the problem of determining if any structural fixed modes are present is equivalent to the standard problem of determining if the system is structurally controllable and observable. It can be verified in this case (e.g. [5]) that the system is structurally controllable and observable, i.e. has no structural fixed modes present.

Example 4': The same example as considered in example 3' is now considered except that the p_1, p_2, p_3, p_4 elements are now constrained to be all equal to each other. In this case, the unspecified system is given by (5) with $p \in R^5$ and

$$C(p) = \begin{pmatrix} p_4 & 0 & 0 & 0 \\ 0 & 0 & p_5 & 0 \end{pmatrix}, \quad A(p) = \begin{pmatrix} 0 & p_1 & 0 & p_1 \\ 0 & 0 & 0 & 0 \\ 0 & p_1 & 0 & p_1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad B(p) = \begin{pmatrix} 0 & 0 \\ p_2 & 0 \\ 0 & 0 \\ 0 & p_3 \end{pmatrix}$$

and it may be easily verified that this system now has a structural fixed mode present.

3. MAIN RESULTS

The following preliminary results are now obtained. Let sp(·) denote the eigenvalues of (·).

Lemma 1: Given $M \in R^{n\times n}$, $N \in R^{n\times n}$, then sp(M) is disjoint from sp(N) if and only if

$$\mathrm{rank}(I_n \otimes M - N' \otimes I_n) = n^2$$

Proof: The proof of this result immediately follows on noting that a solution exists to the matrix equation $MX - XN = I_n$ for X if and only if sp(M) is disjoint from sp(N).

Lemma 2: Consider the unspecified system (5); then

(i) If $\{C(p), A(p), B(p)\}$ has no decentralized fixed modes with respect to $\mathbf{K}$ for some parameter $\bar{p} \in R^{\gamma}$, this implies that $\{C(p), A(p), B(p)\}$ has no structural decentralized fixed modes with respect to $\mathbf{K}$.

(ii) If $\{C(p), A(p), B(p)\}$ has a structural decentralized fixed mode with respect to $\mathbf{K}$, this implies that $\{C(p), A(p), B(p)\}$ has a decentralized mode with respect to $\mathbf{K}$ for all parameters $p \in R^{\gamma}$.

Proof: Consider the proof of (i). Now $\{C(p), A(p), B(p)\}$ for a given $p \in R^{\gamma}$ has no decentralized fixed modes with respect to $\mathbf{K}$ if and only if the eigenvalues of $A(p)$ are disjoint from the eigenvalues of $A(p) + B(p)KC(p)$ for some choice of $K \in \mathbf{K}$, or alternatively from lemma 1, if and only if

$$\mathrm{rank}\{I_n \otimes A(p) - [A(p) + B(p)KC(p)]' \otimes I_n\} = n^2 \tag{7}$$

for some choice of $K \in \mathbf{K}$. Since by assumption there exists $\bar{p} \in R^{\gamma}$ and $\bar{K} \in \mathbf{K}$ so that (7) is satisfied, it immediately follows, by using the same arguments as used in the proof of theorems 1-3 of [7], that this implies that (7) must hold for almost all parameters $p \in R^{\gamma}$ and for almost all $K \in \mathbf{K}$, which proves the result.

The proof of (ii) follows immediately from (i) by using contradiction arguments.

The following result is now immediately obtained from lemma 1.

Theorem 2: Consider the unspecified system (5); then the system $\{C(p), A(p), B(p)\}$ has a structural decentralized fixed mode with respect to $\mathbf{K}$ if and only if:

$$\mathrm{sp}[A(p)] \text{ is not disjoint from } \mathrm{sp}[A(p) + B(p)KC(p)]$$

for all $p \in R^{\gamma}$ and for all $K \in \mathbf{K}$.

The following corollary follows directly from these results and lemma 1.

Corollary 1: Given the unspecified system (5), then:

(i) If $\{C(p), A(p), B(p)\}$ has no structural decentralized fixed modes with respect to $\mathbf{K}$, this implies that:

$$\mathrm{rank}\{I_n \otimes A(p) - [A(p) + B(p)KC(p)]' \otimes I_n\} = n^2$$

and that

$$\mathrm{sp}(A(p)) \text{ is disjoint from } \mathrm{sp}[A(p) + B(p)KC(p)]$$

for almost all $p \in R^{\gamma}$, and almost all $K \in \mathbf{K}$.

(ii) If $\{C(p), A(p), B(p)\}$ has a structural decentralized fixed mode with respect to $\mathbf{K}$, this implies that:

$$\mathrm{rank}\{I_n \otimes A(p) - [A(p) + B(p)KC(p)]' \otimes I_n\} < n^2$$

and that

$$\mathrm{sp}[A(p)] \text{ is not disjoint from } \mathrm{sp}[A(p)+B(p)KC(p)]$$

for all $p \in R^{\gamma}$, and all $K \in \mathbf{K}$.

Corollary 1 forms the basis of the following algorithm to determine if the unspecified system (5) has a structural decentralized fixed mode.

Algorithm I (To determine presence of structural decentralized fixed modes): Given the unspecified system (5),

1. Select an arbitrary $\bar{p} \in R^{\gamma}$ and an arbitrary $\bar{K} \in \mathbf{K}$ (by using a pseudorandom number generator say).
2. Find the eigenvalues of $A(\bar{p})$.
3. Find the eigenvalue of $A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p})$.
4. Then:
   (i) If $\mathrm{sp}(A(\bar{p}))$ is disjoint from $\mathrm{sp}(A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p}))$, the system (5) does not have any structural decentralized fixed modes with respect to $\mathbf{K}$.
   (ii) If $\mathrm{sp}(A(\bar{p}))$ is not disjoint from $\mathrm{sp}(A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p}))$, then for almost all $\bar{p}$, K elements chosen, it can be concluded that the system (5) does have a structural decentralized fixed mode present.

Proof: The proof of step 4 in the algorithm follows directly from Corollary 1.

Remark 2:

(a) If there is any doubt whether the system does have a structural decentralized fixed mode (e.g. in the case of 4(ii) above), the algorithm can be repeated with a different choice of $\bar{p} \in R^{\gamma}$, $\bar{K} \in \mathbf{K}$ made in step 1.

(b) The algorithm could be modified by using the condition

$$\mathrm{rank}\{I_n \otimes A(\bar{p}) - [A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p})]' \otimes I_n\} = n^2$$

in place of the condition that $\mathrm{sp}(A(\bar{p}))$ and $\mathrm{sp}[A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p})]$ be disjoint. This has the advantage of not requiring eigenvalue calculation, but the disadvantage of requiring the determination of the rank of a large matrix $(n^2 \times n^2)$ to be carried out.

(c) In the proposed algorithm, a comparison of the eigenvalues of two matrices is to be made to decide if the matrices have a common eigenvalue. It is important therefore that a method should be used to compute eigenvalues which minimizes numerical error "as much as possible", e.g. the QR transformation method with balancing [8]. Even so, it is possible that a given matrix may be "ill-conditioned" in the sense that its eigenvalues may be very sensitive to changes in the elements of the matrix, e.g. see [11]. Thus it is also important to use in the computation of the eigenvalues of a matrix, the highest precision which is possible, e.g. double precision. When the computed eigenvalues of $A(\bar{p})$ and $A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p})$ are obtained, it is clear that they may in fact be accurate only to an unknown limited number of figures, e.g. 8 significant figures if double precision wordlength is used on an IBM computer. Thus the algorithm requires some subjective decision as to what constitutes a common eigenvalue between two matrices. Some guidance can be given in this regard however, by choosing different $\bar{K}$ matrices for a given $\bar{p}$ vector, and on noting that if (5) has a decentralized structural fixed mode present, then this implies that $A(\bar{p})$, $A(\bar{p})+B(\bar{p})K_i C(\bar{p})$, $i=1,2,3,\ldots$ must all share the same common eigenvalue.

4. NUMERICAL EXAMPLES

A number of examples taken from the literature, and a non-trivial problem, are considered to illustrate the algorithm. All computations were performed on an IBM 3033 II digital computer with double precision word length using the EISPACK package [8] to calculate eigenvalues.

Example 1 [2]: It is desired to determine if the system

$$\dot{x} = \begin{bmatrix} 0 & \times & 0 & \times \\ 0 & 0 & \times & 0 \\ \times & 0 & \times & 0 \\ 0 & \times & 0 & \times \end{bmatrix} x + \begin{bmatrix} \times \\ 0 \\ \times \\ 0 \end{bmatrix} u \tag{8}$$

where × denotes an unspecified element, is structurally controllable [2]. In this case, the problem corresponds to an unspecified system (5) where $p \in R^9$ and $C=I_4$ with $\mathbf{K}=\{(1,1),(1,2),(1,3),(1,4)\}$. On applying Algorithm I, the following arbitrary choice of elements was made by using a pseudorandom number generator (to 6 significant figures);

$$\bar{p} \triangleq \begin{bmatrix} a_{12} \\ a_{14} \\ a_{23} \\ a_{31} \\ a_{33} \\ a_{42} \\ a_{44} \\ b_{11} \\ b_{31} \end{bmatrix} = \begin{bmatrix} -0.818534 \\ -0.852500 \\ -0.445836 \\ 0.059496 \\ 0.881960 \\ 0.540410 \\ -0.749268 \\ -0.968264 \\ 0.736496 \end{bmatrix}$$

$$\bar{K} = (0.472450, 0.450824, 0.998916, 0.777146)$$

This resulted in the following eigenvalues for $A(\bar{p})$, $A(\bar{p})+B(\bar{p})KC(\bar{p})$ (see Table 1) to 8 significant figures:

TABLE 1: Eigenvalues Obtained for Example 1

sp($A(\bar{p})$)	sp($A(\bar{p})+B(\bar{p})KC(\bar{p})$)
0.91653907	1.3669555
-0.76179351	-0.88555034
-0.011026780±j0.20169334	-0.0352359493±j0.41953350

Since the eigenvalues are disjoint, it is concluded that the system is structurally controllable which agrees with [2].

Example 2 [5]: It is desired to determine if the system

$$\dot{x} = \begin{bmatrix} \times & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \times \\ \times & \times & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \times & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \times & 0 & 0 & 0 & 0 & 0 \\ 0 & \times & 0 & \times & \times & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & 0 & \times \end{bmatrix} x + \begin{bmatrix} \times & 0 \\ 0 & 0 \\ 0 & 0 \\ \times & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \times \\ 0 & 0 \\ 0 & 0 \end{bmatrix} u \qquad (9)$$

where $\times$ denotes an unspecified element is structurally controllable [5]. In this case, the problem corresponds to an unspecified system (5) where $p \in R^{18}$, $C = I_9$ and $\mathbf{K} = \{(1,1),(1,2),\ldots,(1,9),(2,1),(2,2),\ldots,(2,9)\}$. On applying Algorithm I and on choosing arbitrary values for $\overline{p} \in R^{18}$ and $K \in R^{2\times 9}$, the following eigenvalues were obtained for $A(\overline{p})$ and $A(\overline{p})+B(\overline{p})KC(\overline{p})$ (see Table 2) to 10 signficant figures:

TABLE 2: Eigenvalues Obtained for Example 2

sp($A(\overline{p})$)	sp($A(\overline{p})+B(\overline{p})KC(\overline{p})$)
-0.9462460000	-1.090018262
-0.9432480000	-0.9827223345
-0.8578720000	-0.7773215686±j0.3178568603
-0.6925600000	-0.3769742538±j0.3942330911
-0.3615000000	0.2217200953
0.1822280000	-0.1393925841
$0.1387778781\times 10^{-16}$	0.01055363615
0.0000000000	
-0.2283700000	

Since the eigenvalues are disjoint, it is concluded that the system is structural controllable which agrees with [5].

Example 3: A model of a large flexible space structure with co-located sensors and actuators can be described as follows [9]:

$$\dot{x} = \begin{bmatrix} 0 & I_n \\ -\Omega & 0 \end{bmatrix} x + \sum_{i=1}^{m} \begin{pmatrix} 0 \\ b_i \end{pmatrix} u_i , \quad x \in R^{2n}$$

$$y_i = (b_i^{'}, b_i^{'})x , \quad i = 1,2,\ldots,m$$

where

$$-\Omega \triangleq \mathrm{diag}\{\underbrace{0,0,\ldots,0}_{s},-\Omega_1,-\Omega_2,\ldots,-\Omega_{n-s}\}$$

and $b_i \in R^n$, $i = 1,2,\ldots,m$. It is desired to determine if this system has any structural decentralized fixed modes for the case when the information flow constraint is given by:

$$\mathbf{K} = \{(1,1),(2,2),\ldots,(m,m)\}$$

This problem corresponds to an unspecified system (5), where $p \in R^{nm+n-s}$ are the parameters associated with the elements of b_i, $i=1,2,\ldots,m$ and the elements Ω_i, $i=1,2,\ldots,n-s$.

In this case, on letting $n=10$, $s=3$, $m=3$ and on choosing arbitrary parameters for $\bar{p} \in R^{37}$, $\bar{K} \triangleq \begin{pmatrix} \bar{k}_1 & 0 & 0 \\ 0 & \bar{k}_2 & 0 \\ 0 & 0 & \bar{k}_3 \end{pmatrix}$, the following eigenvalues were obtained for sp$(A(\bar{p}))$, sp$(A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p}))$ (see Table 6) to 10 significant figures:

TABLE 3: Eigenvalues Obtained for Example 3

sp$(A(\bar{p}))$	sp$(A(\bar{p})+B(\bar{p}KC(\bar{p}))$
0	$-5.383600814\times 10^{-8}\pm j5.288676113\times 10^{-3}$
0	$-2.054650825\times 10^{-5}\pm j4.265900517\times 10^{-1}$
0	$-3.414060465\times 10^{-4}\pm j1.796490771\times 10^{-1}$
0	$-3.716365906\times 10^{-4}\pm j7.941203096\times 10^{-2}$
0	$-8.982089627\times 10^{-4}\pm j7.369355967\times 10^{-1}$
0	$-3.374628866\times 10^{-3}\pm j2.803143364\times 10^{-1}$
$0\pm j0.038350000$	$-8.703570948\times 10^{-3}\pm j3.201888937\times 10^{-1}$
$0\pm j0.2806840000$	$-2.937285285\times 10^{-1}\pm j7.957350441\times 10^{-1}$
$0\pm j0.3052440000$	$-3.776239690\times 10^{-1}\pm j9.592321916\times 10^{-1}$
$0\pm j0.3748300000$	-0.7140594162
$0\pm j0.4727580000$	3.034215011
$0\pm j0.6762040000$	
$0\pm j0.7501100000$	

In this case, since the eigenvalues of the system are disjoint, it is concluded that the system has no structural decentralized fixed modes.

Example 4: This example is identical to example 3, except that now $m=2$. The following eigenvalues for sp$(A(\bar{p}))$, sp$(A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p}))$ were then obtained on choosing arbitrary parameters for $\bar{p} \in R^{27}$, $\bar{K} = \begin{pmatrix} \bar{k}_1 & 0 \\ 0 & \bar{k}_2 \end{pmatrix}$ (see Table 4):

TABLE 4: Eigenvalues Obtained for Example 4

sp$(A(\bar{p}))$	sp$(A(\bar{p})+B(\bar{p})\bar{K}C(\bar{p}))$
0	$1.665334537\times 10^{-16}\pm j1.442798729\times 10^{-8}$
0	$1.837353825\times 10^{-7}\pm j3.591077938\times 10^{-2}$
0	$2.570627976\times 10^{-5}\pm j2.799717705\times 10^{-1}$
0	$4.860368599\times 10^{-5}\pm j4.706465985\times 10^{-1}$
0	$1.100748460\times 10^{-4}\pm j3.403356950\times 10^{-1}$
0	$1.451691901\times 10^{-4}\pm j1.794657747\times 10^{-1}$
$0\pm j.0383500000$	$3.767072513\times 10^{-4}\pm j6.880359533\times 10^{-1}$
$0\pm j.2806840000$	$-7.927766029\times 10^{-3}\pm j5.768626102\times 10^{-1}$
$0\pm j.3052440000$	$-7.639381446\times 10^{-1}\pm j1.059253824$
$0\pm j.3748300000$	-0.6291368400
$0\pm j.4727580000$	1.912692159
$0\pm j.6762040000$	
$0\pm j.7501100000$	

In this case, since there is a common eigenvalue at zero, (assuming that the eigenvalues computed are correct to 7 significant figures say), it is concluded that the system, with $m=2$, has a structural fixed mode of zero, (which corresponds to a rigid body mode of the spacecraft).

5. CONCLUSIONS

This paper has proposed a simple method for deciding if a system, parametrized by unspecified parameters, has a structural decentralized fixed mode present with respect to an arbitrary information flow constraint. The advantage of the method is that it may be applied to a large class of problems, is simple to implement, and can easily be applied to high order systems which have a large number of inputs and outputs. The disadvantage of the method is that some care may be required in interpreting whether two matrices have a common eigenvalue. The method has been applied to problems of 200*th* order with 5 inputs and 5 outputs arising from a large flexible space structure with no difficulty.

REFERENCES

[1] Wang, S.H., Davison, E.J., "On the stabilization of decentralized control systems", *IEEE Trans. on Automatic Control*, vol. AC-18, 1973, pp. 473-478.

[2] Lin, C.T., "Structural controllability", *IEEE Trans. on Automatic Control", 1974, pp. 201-208.*

[3] Shields, R.W., Pearson, J.B., "Structural controllability of multi-input linear systems", *IEEE Trans. on Automatic Control*, vol. AC-21, 1976, pp. 203-212.

[4] Glover, K., Silverman, L.M., "Characterization of structural controllability", *IEEE Trans. on Automatic Control*, vol. AC-21, 1976, pp. 534-537.

[5] Davison, E.J., "Connectability and structural controllability of composite systems", *Automatica*, vol. 13, 1977, pp. 109-123.

[6] Sezer, M.E., Siljak, D.D., "Structurally fixed modes", *Systems and Control Letters*, vol. 1, no. 1, July 1981, pp. 60-64.

[7] Davison, E.J., Wang, S.H., "Properties of linear multivariable systems subject to arbitrary output and state feedback", *IEEE Trans. on Automatic Control*, vol. AC-18, no. 1, 1973, pp. 24-32.

[8] Smith, B.T., et al., *Matrix Eigensystem Routines - EISPACK Guide.* Berlin: Springer-Verlag, 1976.

[9] West-Vukovich, G., Davison, E.J., Hughes, P., "The decentralized control of large flexible space structures", *IEEE Trans. on Automatic Control*, vol. AC-29, no. 10, 1984, pp. 866-879.

[10] Davison, E.J., "Decentralized stabilization and regulation in large multivariable systems", in *Directions in Decentralized Control, Many Person Optimization and Large Scale Systems*, (editors: Y.C. Ho, S. Mitter), Plenum Press, 1976, pp. 303-323.

[11] Wilkinson, J.H., Reinsch, C. *Linear Algebra.* Springer-Verlag, 1971.

[12] Davison, E.J., Wang, S.H., "A characterization of decentralized fixed modes in terms of transmission zeros", *IEEE Trans. on Automatic Control*, vol. AC-30, no. 1, 1985, pp. 81-82.

DECENTRALIZED LQG/LTR CONTROLLER DESIGN WITH LOCAL MODELS

A. İFTAR and Ü. ÖZGÜNER
Department of Electrical Engineering
The Ohio State University
2015 Neil Avenue
Columbus, OH 43210

ABSTRACT

The LQG/LTR Design methodology for decentralized control systems is considered. Bounds on the norm of the multiplicative error matrix are calculated. It is shown that the controllers can be designed locally by using local models.

INTRODUCTION

The linear-quadratic-Gaussian with loop-transfer-recovery (LQG/LTR) design methodology has been recently introduced by Doyle and Stein [1] and has attracted much interest since then (e.g., [2]-[3]). This methodology enables frequency domain controller design by using the well known LQG techniques.

In the present paper we consider the LQG/LTR methodology for decentralized control systems. We specifically calculate bounds on the norm of the *multiplicative error matrix* (given in (13)) and use the LQG/LTR methodology to design local controllers. The *design model* (9) and the local models are developed with the method of *expansion* introduced by Šiljak and co-workers [4].

DEVELOPMENTS

Consider a system with two decentralized control agents described by

$$\dot{x} = Ax + B_1u_1 + B_2u_2, x(0) = x_0 \epsilon R^n \quad (1a)$$

$$y_1 = C_1x \quad (1b)$$

$$y_2 = C_2x \quad (1c)$$

where $x \epsilon R^n$ is the state, $u_i \epsilon R^{m_i}$ is the input, and $y_i \epsilon R^{m_i}$ is the output of the ith control agent ($i = 1,2$). It is assumed that the system given above does not have any decentralized fixed modes (DFM's) [5].

Consider the following partition of the state x:

$$x = \begin{bmatrix} x_{o1} \\ x_{oc1} \\ x_{oc} \\ x_{oc2} \\ x_{o2} \end{bmatrix} \tag{2}$$

where $x_{oi} \epsilon R^{n_{oi}}$ is observable only by agent-i, $x_{oci} \epsilon R^{n_{oci}}$ is observable by both agents, but controllable only by agent-i (i =1,2), and $x_{oc} \epsilon R^{n_{oc}}$ is observable and controllable by both agents. Note that the assumption of no DFM's implies that x_{oi} is controllable by agent-i (i = 1,2). Then the matrices of (1) have the following forms [6]:

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & 0 \\ 0 & A_{22} & 0 & 0 & 0 \\ 0 & A_{32} & A_{33} & A_{34} & 0 \\ 0 & 0 & 0 & A_{44} & 0 \\ 0 & A_{52} & A_{53} & A_{54} & A_{55} \end{bmatrix} \tag{3a}$$

$$B \triangleq [B_1 \ : \ B_2] = \begin{bmatrix} B_{11} & : & B_{12} \\ B_{21} & : & 0 \\ B_{31} & : & B_{32} \\ 0 & : & B_{42} \\ B_{51} & : & B_{52} \end{bmatrix} \tag{3b}$$

$$C \triangleq \begin{bmatrix} C_1 \\ \cdots \\ C_2 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & C_{22} & C_{23} & C_{24} & C_{25} \end{bmatrix} \tag{3c}$$

Consider the transformation:

$$\hat{x} = Tx = \begin{bmatrix} x_{o1} \\ x_{oc1} \\ x_{oc} \\ \cdots \\ x_{oc} \\ x_{oc2} \\ x_{o2} \end{bmatrix} \epsilon R^{\hat{n}}, \quad \hat{n} = n + n_{oc} \tag{4}$$

The expansion [4] of the original system (1) with respect to the transformation (4) is given by (see [7] for developments):

$$\dot{\hat{x}} = \hat{A}\hat{x} + \hat{B}u, \quad \hat{x}(0) = Tx_0 \epsilon R^{\hat{n}} \tag{5a}$$

$$y = \hat{C}\hat{x} \tag{5b}$$

where

$$\hat{A} = \left[\begin{array}{ccccccc} A_{11} & A_{12} & A_{13} & \vdots & 0 & A_{14} & 0 \\ 0 & A_{22} & 0 & \vdots & 0 & 0 & 0 \\ 0 & A_{32} & A_{33} & \vdots & 0 & A_{34} & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & A_{32} & 0 & \vdots & A_{33} & A_{34} & 0 \\ 0 & 0 & 0 & \vdots & 0 & A_{44} & 0 \\ 0 & A_{52} & 0 & \vdots & A_{53} & A_{54} & A_{55} \end{array}\right] \triangleq \begin{bmatrix} \hat{A}_1 & \hat{A}_{12} \\ \hat{A}_{21} & \hat{A}_2 \end{bmatrix} \tag{6a}$$

$$\hat{B} = \left[\begin{array}{ccc} B_{11} & \vdots & B_{12} \\ B_{21} & \vdots & 0 \\ B_{31} & \vdots & B_{32} \\ \cdots & \cdots & \cdots \\ B_{31} & \vdots & B_{32} \\ 0 & \vdots & B_{42} \\ B_{51} & \vdots & B_{52} \end{array}\right] \triangleq \begin{bmatrix} \hat{B}_1 & \hat{B}_{12} \\ \hat{B}_{21} & \hat{B}_2 \end{bmatrix} \tag{6b}$$

$$\hat{C} = \left[\begin{array}{ccccccc} C_{11} & C_{12} & C_{13} & \vdots & 0 & C_{14} & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & C_{22} & 0 & \vdots & C_{23} & C_{24} & C_{25} \end{array}\right] \triangleq \begin{bmatrix} \hat{C}_1 & \hat{C}_{12} \\ \hat{C}_{21} & \hat{C}_2 \end{bmatrix} \tag{6c}$$

It can be shown that [4], the response $x(t)$ of (1) is related to the response $\hat{x}(t)$ of (5) by

$$x(t) = T^I \hat{x}(t) \quad \forall t \geq 0 \tag{7}$$

where T^I is the generalized inverse of T satisfying $T^I T = I$. Furthermore (1) and (5) have equivalent input/output descriptions, i.e.,

$$\hat{G}(s) \triangleq \hat{C}(sI - \hat{A})^{-1}\hat{B} \equiv G(s) \triangleq C(sI - A)^{-1}B \tag{8}$$

Now consider the *design model* described by:

$$\dot{\tilde{x}} = \tilde{A}\tilde{x} + \tilde{B}u \tag{9a}$$

$$y = \tilde{C}\tilde{x} \tag{9b}$$

where

$$\tilde{A} = \begin{bmatrix} \hat{A}_1 & 0 \\ 0 & \hat{A}_2 \end{bmatrix}, \tag{10a}$$

$$\tilde{B} = \begin{bmatrix} \hat{B}_1 & 0 \\ 0 & \hat{B}_2 \end{bmatrix}, \tag{10b}$$

$$\tilde{C} = \begin{bmatrix} \hat{C}_1 & 0 \\ 0 & \hat{C}_2 \end{bmatrix} \tag{10c}$$

The transfer function matrix (TFM) for the design model (9) is given by:

$$\tilde{G}(s) \triangleq \tilde{C}(sI - \tilde{A})^{-1}\tilde{B} = \text{ block diag } (G_1(s), G_2(s)) \tag{11}$$

where $G_i(s)$ is the TFM for the decoupled subsystem:

$$\dot{x}_i = \hat{A}_i x_i + \hat{B}_i u_i \tag{12a}$$

$$y_i = \hat{C}_i x_i, \qquad i = 1,2 \tag{12b}$$

Note that (12) is the minimal local model for agent-i.

Let $\hat{G}(s)$ and $\tilde{G}(s)$ be related by

$$\hat{G}(s) = \tilde{G}(s)(I + E(s)) \tag{13}$$

where $E(s)$ is the so-called *multiplicative error matrix* between the true TFM $G(s)$ (or equivalently $\hat{G}(s)$) and the *design TFM* $\tilde{G}(s)$. An upper bound on the norm of $E(j\omega)$ can be found as:

$$\bar{\sigma}(E(j\omega)) \le \frac{\bar{\sigma}(\hat{G}(j\omega) - \tilde{G}(j\omega))}{\underline{\sigma}(\tilde{G}(j\omega))} \triangleq e_1(\omega) \tag{14}$$

where $\bar{\sigma}(\cdot)$ and $\underline{\sigma}(\cdot)$ indicate, respectively, the maximum and the minimum singular values of the indicated matrices.

If any part of the system (1) that is observable by agent-i is also controllable by agent-i, then $n_{oci} = 0 (i = 1,2)$ and a more simplified bound

$$\bar{\sigma}(E(j\omega)) \le \max\left(\frac{\bar{\sigma}(\hat{B}_1^T \hat{B}_{12})}{\underline{\sigma}(\hat{B}_1^T \hat{B}_1)}, \frac{\bar{\sigma}(\hat{B}_2^T \hat{B}_{21})}{\underline{\sigma}(\hat{B}_2^T \hat{B}_2)}\right) \triangleq e_2(j\omega) \tag{15}$$

can be found.

From (14) and (15) it can be deduced that if the interactions between two subsystems are *weak*, then the *size* of the error matrix $E(s)$ is *small*.

Consider the local target feedback loop (TFL) shown in Fig. 1. The control gains K_i ($i = 1,2$) are determined by considering a linear quadratic regulator problem and are given by:

$$K_i = ({}^1/\rho_i)\hat{B}_i^T P_i, \quad i = 1,2 \tag{16}$$

where $P_i = P_i^T > 0$ is the solution of:

$$\hat{A}_i^T P_i + P_i \hat{A}_i + L_i^T L_i - ({}^1/\rho_i) P_i \hat{B}_i \hat{B}_i^T P_i = 0 \tag{17}$$

The *design parameters* ρ_i and $L_i \epsilon R^{m_i \times n_i}$ are chosen such that the stability-robustness requirement [1]:

$$\bar{\sigma}(T(j\omega)) < \frac{1}{e_m(\omega)}, \quad \forall \omega \tag{18}$$

and the performance criterion (e.g., see [1]) are satisfied. Here

$$T(s) \triangleq (I + G_K(s))^{-1} G_K(s) \tag{19}$$

is the closed-loop TFM for the TFL,

$$G_K(s) \triangleq K(sI - \bar{A})^{-1} \bar{B} \tag{20}$$

is the loop TFM, and

$$K \triangleq \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix} \tag{21}$$

If there are no modelling uncertainties in the model (1), then $e_m(\omega)$ in (18) can be chosen as $e_1(\omega)$ (given by (14)) or $e_2(\omega)$ (given by (15)), to account for the *interactions* between two subsystems. However, if there are modelling uncertainties, then they must also be reflected in $e_m(\omega)$, e.g., by choosing

$$e_m(\omega) = e_i(\omega) e_c(\omega) + e_o(\omega), \qquad (i = 1 \text{ or } 2) \tag{22}$$

where $e_c(\omega) \geq 1$ accounts for modeling uncertainties in interactions and $e_o(\omega) \geq 0$ accounts for uncertainties in subsystem models.

In choosing ρ_i and L_i ($i = 1,2$) the following identity (for $\rho = \rho_1 = \rho_2$) [1] is particularly useful:

$$\sigma_j(I + G_K(j\omega)) = \sqrt{1 + \frac{1}{\rho}\sigma_j^2[L(j\omega - \tilde{A})^{-1}\tilde{B}]}\,, \quad j = 1, \ldots, \hat{n} \tag{23}$$

where $\sigma_j(\cdot)$ is the jth singular value of the indicated matrix. However, note that due to the block diagonal nature of the design model (9), ρ_i and $L_i (i = 1,2)$ can be chosen locally by considering

$$\sigma_j(I + G_{K_i}(j\omega)) = \sqrt{1 + \frac{1}{\rho_i}\sigma_j^2[L_i(j\omega - \tilde{A}_i)^{-1}\tilde{B}_i]}, \quad j = 1, \ldots, n_i, i = 1,2 \tag{24}$$

where

$$G_{K_i}(s) \triangleq K_i(sI - \hat{A}_i)^{-1}\hat{B}_i, \quad i = 1,2 \tag{25}$$

Furthermore, since the stability-robustness requirement (18) can be stated as:

$$\bar{\sigma}(T_i(j\omega)) < \frac{1}{e_m(\omega)}, \quad \forall\, i = 1,2, \quad \forall \omega \tag{26}$$

where

$$T_i(s) \triangleq (I + G_{K_i}(s))^{-1}G_{K_i}(s), \quad i = 1,2 \tag{27}$$

this requirement can also be checked locally.

If $(\hat{A}_i, \hat{B}_i, \hat{C}_i), \quad i = 1,2$ have no non-minimum phase zeros, then the TFL performance can be recovered by the following local modified Kalman filter designs as $\mu_i \rightarrow 0$ [1]:

$$H_i = ({}^1\!/\mu_i)\Pi_i\hat{C}_i^T, \quad i = 1,2 \tag{28}$$

where $\Pi_i = \Pi_i^T > 0$ is the solution of:

$$\hat{A}_i\Pi_i + \Pi_i\hat{A}_i^T + \hat{B}_i\hat{B}_i^T - ({}^1\!/\mu_i)\Pi_i\hat{C}_i^T\hat{C}_i\Pi_i = 0 \tag{29}$$

and the local controllers are described by

$$u_i = -K_i z_i \tag{30a}$$
$$\dot{z}_i = (\hat{A}_i - \hat{B}_i K_i - H_i\hat{C}_i)z - H_i e_i \tag{30b}$$
$$e_i = r_i - y_i, \qquad i = 1,2 \tag{30c}$$

The resulting closed-loop system is shown in Fig. 2. This system is guaranteed to stay stable as long as (18) or equivalently (26) is satisfied.

CONCLUSIONS

Decentralized LQG/LTR design methodology has been studied. Upper bounds on the norm of the multiplicative error matrix, that accounts for the interactions between the subsystems, has been calculated. It has been shown that the design process can be carried out locally once the above mentioned bound is known by all control agents. Note that the design process also includes checking for stability-robustness requirement (26) and performance criteria [1].

Although only decentralized systems with two control agents have been considered for notational simplicity, the extension of the results to systems with more than two control agents is straightforward.

REFERENCES

[1] J. C. Doyle and G. Stein, "Multivariable feedback design concepts for a classical/modern synthesis," *IEEE Trans. Aut. Contr.*, vol. AC-26, pp. 4-16, 1981.

[2] M. Athans, "A tutorial on the LQG/LTR method," *Proc. American Contr. Conf.*, Seattle, WA, pp. 1289–1295, 1986.

[3] H. H. Yeh, S.S. Banda, P. J. Lynch, and T. E. McQuade, "The LQG/LTR design via H_2-optimization," *Proc. American Contr. Conf.*, Seattle, WA, pp. 2097–2103, 1986

[4] M. Ikeda, D. D. Šiljak, and D. E. White, "Decentralized control with overlapping information sets," *J. Opt. Theory and Applic.*, vol. 34, pp. 279–310, 1981.

[5] S. H. Wang and E. J. Davison, "On the stabilization of decentralized control systems," *IEEE Trans. Aut. Contr.*, vol. AC-18, pp. 473–478, 1973.

[6] Ü. Özgüner and W. R. Perkins, "Structural properties of large-scale composite systems," in *Large-Scale Dynamical Systems*, Point Lobos Press, N. Hollywood, CA, R. Seaks, ed., 1975.

[7] A. İftar and Ü. Özgüner, "Closed-loop balanced realizations in the analysis of suboptimality and stability of decentralized systems," *Proc. IEEE Conf. Dec. Contr.*, Las Vegas, NV, pp. 143–148, 1984.

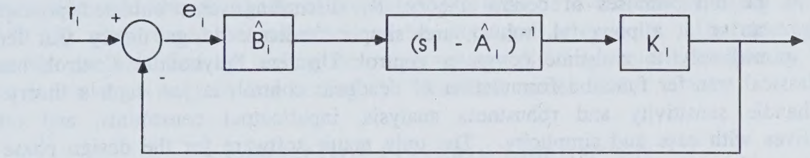

r
1
+
e
1
-
B̂
1
(s I - Â
1
)
K
1

Figure 1: Local TFL

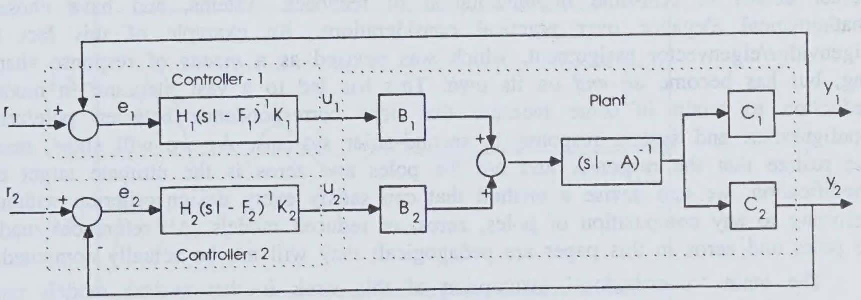

Controller - 1
r
1
+
e
1
H
1
(s I - F
1
)
-1
K
1
u
1
B
1
Plant
+
(s I - A)
-1
C
1
y
1
r
2
+
e
2
-
H
2
(s I - F
2
)
-1
K
2
u
2
B
2
+
C
2
y
2
Controller - 2

Figure 2: Closed-loop control system [$F_i \triangleq \hat{A}_i - \hat{B}_i K_i - H_i \hat{C}_i, \quad i = 1, 2$].

DISCRETE POLYNOMIAL CONTROL THEORY

KAMRAN OSSIA and H. W. SMITH

University of Toronto
Department of Electrical Engineering
10 King's College Road
Toronto, Ont. Canada M5S 1A4

ABSTRACT

Recent advances in computing and microprocessor technology warrant a new look at the old premises of control theory. By discarding some outdated principles one can arrive at a powerful, robust, and simple feedback design theory that lends itself immediately to real-time computer control. Discrete Polynomial Control, based on classical transfer function formulation of deadbeat control, is just such a theory. It can handle sensitivity and robustness analysis, input/output constraints, and other objectives with ease and simplicity. The only major software for the design phase is singular value decomposition and any 8-bit microprocessor suffices to control complex dynamic systems.

1. INTRODUCTION

Despite its great advances in theory, modern control has not enjoyed widespread use in industrial applications. It seems that most authors are unfamiliar with the actual details of real-time implementation of feedback systems, and have chosen mathematical elegance over practical considerations. An example of this fact is eigenvalue/eigenvector assignment, which was devised as a *means* of response shaping, but has become an *end* on its own. This has led to a vast literature in model reduction to retain in some measure the close correspondence between pole/zero configuration and system response in second-order systems. As we will show, once we realize that the response, and not the poles and zeros is the ultimate target of modification, we can devise a method that can satisfy every design criterion without resorting to any computation of poles, zeros, or reduced models. All references made to poles and zeros in this paper are pedagogical; they will not be actually computed.

The main ''unorthodox'' assumption of this work is that system models and dynamic controllers may be of arbitrarily high order. No attempt will be made in model reduction or designing low-order controllers. Also no pole placement will be considered; the system will be deadbeat at all times. Finally, since only sampled time responses are used, continuous and discrete systems are treated identically.

With these assumptions, the task of control system design is carried out in three stages. First a matrix fraction description is obtained for the system using Principal Hankel Component identification [2,3]. Second, an open-loop controller is designed to give the system a deadbeat response and to satisfy the input/output magnitude and rate constraints. Third, a closed-loop controller is chosen which produces the same overall response as the open-loop system.

2. PRINCIPAL HANKEL COMPONENT IDENTIFICATION

The properties of finite and infinite Hankel matrices have been studied for many years, and their application in system identification are well-known [2,3,4]. Recently, two methods called the Principal Hankel Component (PHC) and Optimal Hankel Norm (OHN) have been proposed for identifying discrete dynamic models of multivariable systems. PHC is simpler and more suited to our purpose, and it gives results nearly identical to OHN for low error thresholds, i.e. if we are not doing any model reduction [3]. We shall briefly outline the PHC method here.

Assuming that an upper limit on the order of the system is known, a Hankel matrix of suitable dimensions is formed from the pulse response of the system. Singular value decomposition is then used to identify both the order and the state-space matrices. This method can easily be modified to yield the matrix fraction description, and the drawback of truncated data can be overcome by using recursive SVD [5] when the pulse response can be computed analytically. Unstable systems can also be identified by scaling the response (say dividing the i-th sample by 2^i) so that it converges to zero and scaling the identified model back.

3. OPEN-LOOP CONTROLLER DESIGN

The main motivation for Discrete Polynomial Control came from the following observation: for a first- or second-order system one can find an open-loop controller with only one switchpoint, i.e. one whose unit pulse response consists of two consecutive pulses, such that it produces a deadbeat response in the system output. This is obvious for a single exponential; for a second-order system with the step response:

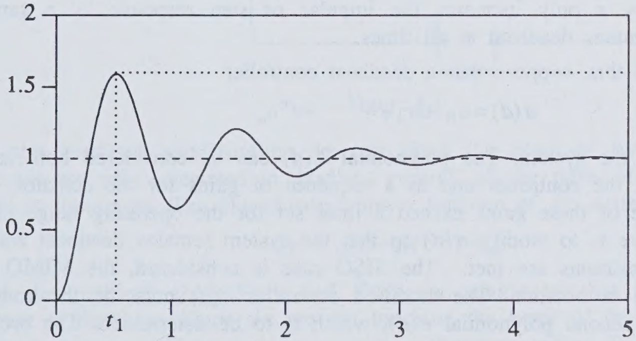


Figure 3.1

we can show that the portion of the response after t_1 is a multiple of the total response plus a constant. Therefore a controller which has a switchpoint at t_1 can be found with the appropriate gains to produce an exactly deadbeat response after t_1. This can be extended to systems of any order, i.e. for any n-th order system a discrete compensator with n switchpoints can be found to give a deadbeat response. The proof of this is directly related to the fact that any n-th order continuous system has an n-th order discrete equivalent.

The second step in our design procedure is finding an open-loop controller which produces a deadbeat response that satisfies the input and output magnitude and rate constraints. Throughout this paper we shall use the unit delay operator d to represent systems.

Suppose that the system is represented by a right MFD as shown in the diagram below:

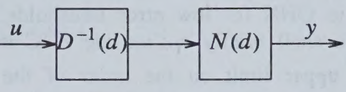


Figure 3.2

The controller which produces deadbeat response in the output is simply D, which in effect cancels the poles of the system and gives an overall impulse response of

$$y = N(d)\, y_{ref}$$

By scaling the matrix $N(d)$, we can accommodate the design constraints. Usually, the reference input is chosen as a sequence of step changes, thus the output is required to follow step inputs without steady state error. This is achieved by scaling N such that $N(1)=I$, the identity matrix. This is in fact a frequency-domain scaling in that the step function corresponds to the eigenvalue at $d=1$. Other classes of reference inputs can be included similarly, although the order of the controller will increase by one for complex roots. Satisfying input/output constraints is achieved by "stretching and flattening" the deadbeat response, i.e. increasing the order of the controller until the response falls within the prescribed limits. Note that increasing the system order by n only increases the impulse or step response by n samples since the system remains deadbeat at all times.

To demonstrate this, suppose that a deadbeat controller

$$a(d) = a_0 + d a_1 + \cdots + d^n a_n$$

has been found for the system. The polynomial $a(d)$ can be considered both as the impulse response of the controller and as a sequence of gains for the actuator. Suppose now that some of these gains exceed a limit set for the operating range of the system. The objective is to modify $a(d)$ so that the system remains deadbeat and the input magnitude constraints are met. The SISO case is considered, the MIMO case being different only in notation. The modified controller $\hat{a}(d)$ must be the convolution of $a(d)$ and a second polynomial $c(d)$, which is to be determined. The problem can be formulated as:

minimize $\max\limits_{i} |\hat{a}_i|$

where $\hat{a}_i = \sum\limits_{j=\alpha}^{\min(i,p)} a_j c_{i-j}, \quad \alpha = \max(0, i-n); \; i = 0, 1, \cdots, p+n$

subject to $\sum\limits_{i=0}^{p} c_i = 1.$

The degree of $c(d)$, i.e. p is not known beforehand, but should be kept to a minimum. The problem as formulated above is a nonlinear one, involving the ∞-norm of a vector. If we replace this with the 2-norm, we will have a linear least-squares problem, at the expense of maybe a few extra elements in $c(d)$, but a much simpler solution. In fact, a closer examination of the problem shows that the 2-norm and the ∞-norm of a vector are close when there is a large disparity among its elements, i.e. when the magnitude constraints are more likely to be violated.

Again we will not mention the details of the solution since it is a simple least squares procedure.

Rate constraints can be handled in a similar fashion, using the differential of the controller polynomial. Of course the two sets of constraints are not unrelated; solving one problem may solve or alleviate the other. Ouput constraints can also be handled in a similar fashion, using the impulse response of the overall system instead of the controller.

The main point here is that with a discrete open-loop controller one has great freedom in shaping the response of the system to conform with any desired specifications.

4. CLOSED-LOOP CONTROLLER DESIGN

In this section we shall try to find a closed-loop controller which is derived from, and gives the same result as the open-loop controller.

We start with a general feedback configuration which is equivalent to state estimation and feedback [6]:

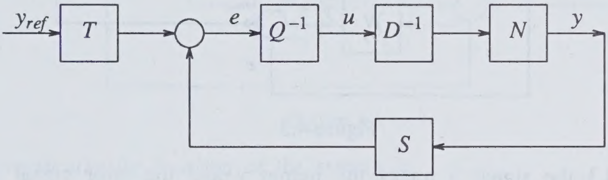


Figure 4.1

With this feedback configuration we can place the poles of the system arbitrarily. Since we are only interested in deadbeat control, all the poles of the system will be placed at the origin. The closed-loop transfer function of the system is

$$y = N(QD+SN)^{-1}T\,y_{ref}$$

where the matrices $Q(d), S(d)$, and $T(d)$ are sought so that the overall impulse response is deadbeat. Since we are not touching the zeros of the system, the desired deadbeat response will have to be $y = N\,y_{ref}$. Therefore setting $T=I$ and finding polynomial matrices such that

$$QD+SN=I$$

will give the same result as the feedforward controller D.

There exists a unity-feedback setup which achieves the same result:

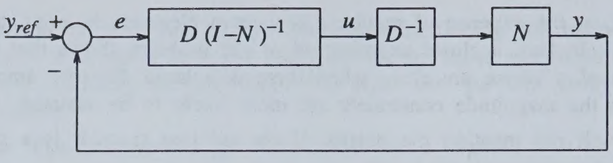


Figure 4.2

It is easy to show that this system has the same impulse response $N(d)$, and furthermore there is no need to solve any polynomial equations. The controller $D(I-N)^{-1}$ has an interesting property; taking the case of constant setpoints, for example, we had already forced the matrix $N(1)$ to be the identity matrix, therefore $I-N(1)$ will be zero. This means that there are m integrators in the controller, and in fact it is a form of proprotional plus integral control. It has the added advantage, however, of being able to accommodate any class of setpoints by simply scaling the model as mentioned above. We will also show that it has the robustness property.

The controller can be implemented in a slightly different way which reveals why the closed-loop system has the same order as $N(d)$:

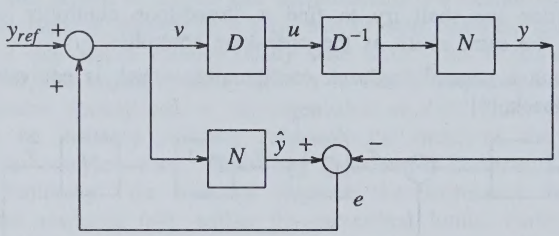


Figure 4.3

In figure 4.3 the signal $\hat{y}$ tracks the output y and the error signal e is identically zero except for initial conditions and disturbances (including errors due to inaccuracies in the model). Therefore no signal originating from y_{ref} will appear in e. In other words, as far as the setpoint is concerned, this is an open-loop system. The feedback loop only serves the purpose of "flushing" the initial conditions in finite time, and stabilizing the system if the original system was unstable. In this way we gain the benefits of both open- and closed-loop design.

Note that the magnitude and rate constraints for both inputs and outputs are assumed to have been incorporated in the D and N matrices during the open-loop design stage.

5. SENSITIVITY AND ROBUSTNESS ANALYSIS

The robustness of feedback control systems is an important factor in design, and much effort has been devoted to this topic for many years. The majority of papers in the literature deal with state-space models and eigenvalue/eigenvector sensitivity. In practice, however, systems with significantly different eigenvalues and eigenvectors can have very similar behavior [1], as is witnessed in the numerous examples of model reduction. Perhaps a better criterion for measuring the sensitivity of feedback systems is the deviation of the closed-loop impulse or step response as a function of the deviation in the open-loop impulse response. In Discrete Polynomial Control, sensitivity and robustness of feedback systems can be analyzed in a straightforward way, with the closed-loop transfer function being an explicit function of the deviation in the open-loop impulse response. In this context, the stability and the tracking and regulation ability of the system can be analyzed easily.

To begin we will assume that a matrix fraction description

$$\hat{H}(d) = \hat{N}(d)\hat{D}^{-1}(d)$$

has been found for the system which differs from its true impulse response $H(d)$. Let $\Delta(d)$ represent the deviation in the response:

$$\Delta(d) \triangleq H(d) - \hat{H}(d)$$

Figure 5.1 is the usual feedback configuration for Discrete Polynomial Control, with the signal w representing the disturbances and the initial conditions.

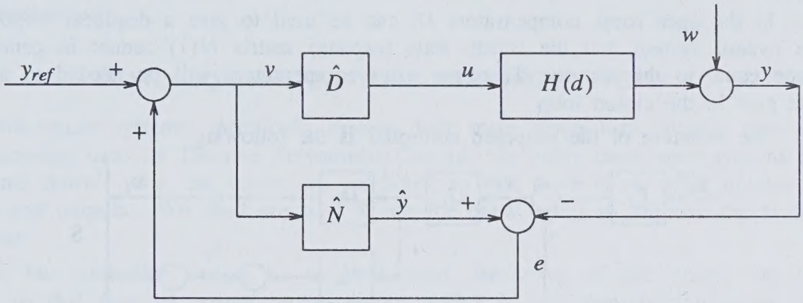


Figure 5.1

The overall transfer function of the system is

$$y = (\hat{N} + \Delta\hat{D})(I + \Delta\hat{D})^{-1} y_{ref} + (I - \hat{N})(I + \Delta\hat{D})^{-1} w$$

Since the matrix $\hat{N}$ found for the system has the property $\hat{N}(1) = I$, even though it may not be an exact representation of the true matrix N, for constant setpoints and disturbances we will still have asymptotic tracking and regulation as long as the system remains stable.

The stability of the closed-loop system can be analyzed in terms of the error sequence $\Delta(d)$: as long as the roots of the polynomial matrix $I - \Delta(d)\hat{D}(d)$ remain outside the unit circle the system will be stable. For any given error sequence it is an easy matter to determine whether the system is stable by directly calculating the inverse sequence of $I - \Delta\hat{D}$. For an exact model, i.e. $\Delta(d) \equiv 0$, we will have deadbeat tracking and regulation:

$$y = N\, y_{ref} + (I - N)w$$

The advantage of this method for sensitivity analysis is that no matter what causes the deviation of the system response, the stability of the closed-loop system can be determined easily, especially since the impulse response can be calculated from any other system description.

6. DECENTRALIZED CONTROL

To design a decentralized discrete polynomial controller, we consider the following model:

$$\mathbf{y}_i = \sum_{j=1}^{m} N_{ij} D_j^{-1} \mathbf{u}_j + \mathbf{w}_i \ , \quad i=1, \cdots , m$$

The objective is to design m controllers, each acting on the inputs and measuring the outputs of one station only, such that the outputs track a corresponding reference input $\mathbf{y}_i^{ref}$ and reject all other setpoints and all disturbances asymptotically. The model given above, i.e. the block column realization, can always be obtained for any system.

In general it is not possible to obtain a closed-loop deadbeat controller for a decentralized system, so a compromise must be found to minimize the transient response.

In the open loop, compensators D_i can be used to give a deadbeat response to the overall system, but the steady state response matrix $N(1)$ cannot in general be made equal to the identity. Therefore extra compensators will be needed to achieve that goal in the closed loop.

The structure of the proposed controller is the following:

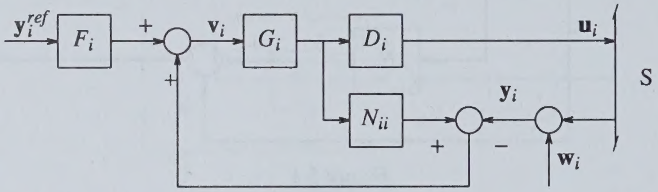


Figure 6.1

where all matrices are polynomial and we define

$$F \triangleq block\ diag(F_1, \cdots ,F_m); \ F(1)=I$$

$$G \triangleq block\ diag(G_1, \cdots ,G_m); \ G(1)=I$$

which are to be determined, and

$$\hat{N} = block\ diag(N_{11}, \cdots ,N_{mm}); \ \hat{N}(1)=I$$

$$D = block\ diag(D_1, \cdots ,D_m)$$

The overall output of the system can be written in terms of the overall reference and disturbance vectors as

$$\mathbf{y} = NG(I+(N-\hat{N})G)^{-1} F \mathbf{y}_{ref} + (I-\hat{N}G)(I+(N-\hat{N})G)^{-1} \mathbf{w}$$

Therefore the matrix G must be found such that $I+(N-\hat{N})G$ is stable. This can always be done owing to the fact that G affects only the off-diagonal elements of the matrix. The solution involves a least squares procedure and is essentially the same as the scaling procedure described in section 3, and will be described in detail elsewhere. By G we can choose the rate of decay of the disturbance rejection which in general will not be deadbeat. The tracking of reference inputs, however, can be made deadbeat by adding an extra set of compensators F_i which are the denominator matrices of the columns of $(I+(N-\hat{N})G)^{-1}$.

To accommodate sinusoidal or other types of setpoints and disturbances, we can replace the conditions above by $F(\lambda)$ etc. where λ is the corresponding mode.

To examine the robustness of this control system, suppose that the approximate values D_e and N_e have been found for D and N, and define

$$\Delta(d) \triangleq ND^{-1} - \hat{N}_e D_e^{-1}$$

We have

$$\mathbf{y} = (\hat{N}_e G + \Delta D_e G)(I+\Delta D_e G)^{-1} F \mathbf{y}_{ref} + (I - \hat{N}_e G)(I+\Delta D_e G)^{-1}\mathbf{w}$$

where $\hat{N}_e$ consists of the diagonal blocks of N_e. Now since $\hat{N}_e(1)G(1) = I$, as long as the closed-loop system remains stable, asymptotic tracking and regulation occurs regardless of the values of Δ and D_e. We have thus designed a robust decentralized servomechanism.

7. NON-SQUARE AND TIME-VARYING SYSTEMS

Non-square systems, specifically systems with more inputs than outputs, represent an interesting case for Discrete Polynomial Control. In many cases such systems are "squared down" , i.e. the inputs are combined so that there is an equal number of inputs and outputs. We shall see how to use the extra inputs to improve the system response.

In our controller design, as we mentioned, the zeros of the system are left alone, so that they all appear in the output, which is why nonminimum-phase systems need no special treatment. Non-square systems on the other hand have no generic zeros except for possible delays, such that the matrix fraction description has a *polynomial* pseudoinverse, which can be calculated directly during the identification process. Using this polynomial as our controller an immediate (more realistically, a single-step) deadbeat response in the output. There is no need for squaring down the system, and all the other design criteria can be met as before.

For time-varying systems, once the matrix fraction descriptions have been identified at the different operating points, the control software can switch between them instantly, assuming of course that the variations in the parameters are slow compared to the system dynamics. Considering the amount of memory available on microprocessors today, there is virtually no limit on the size of systems and number of operating points that can be accommodated by this method.

8. IMPLEMENTATION

The structure of the discrete polynomial controller is quite suited to microprocessor implementation. The program is very simple and includes little more than calculating the inner product of vectors. We shall present the single-input single-output version in "C"; the multi-input multi-output one is different only in the array dimensions and the number of loops.

Following Figure 4.3, for a strictly proper system we have

$$u = Dv$$

$$v = y_{ref} - y + Nv$$

where

$$N = N_1 d + N_2 d^2 + \cdots + N_p d^p$$
$$D = D_0 + D_1 d + \cdots + D_q d^q$$

Denoting real time samples by superscripts, we have

$$v^k = y_{ref}^k - y^k + N_1 v^{k-1} + N_2 v^{k-2} + \cdots + N_p v^{k-p}$$

$$u^k = D_0 v^k + D_1 v^{k-1} + \cdots + D_q v^{k-q}$$

The program implementing the equations above is the following. Here we have assumed that 256 levels are enough to represent the inputs and outputs, giving a quantization error of less than 0.5%. Thus the values are passed to the program as eight bit characters. It is a simple matter to change this to a 16 bit word representing 65535 levels. Also the scalings of the inputs and output are omitted because all equations are linear and they would cancel out. The values of v[0] to v[p] represent v^k to v^{k-p}.

```
#include <stdio.h>
dpc( )
/**/
/* Discrete Polynomial Controller for a SISO system */
/* p, q are Numerator and Denominator Degrees; p >= q  */
/* N(p) and D(q) are known vectors */
/* f1 and f2 are input streams; f3 is the output stream */
/* atoi: ASCII to integer conversion */
/* dtoa: double to ASCII conversion */
/**/
int p, q, y, yref, i
double N[ ], D[ ], v[ ], t, u
{ for ( ; ; ) {
                y = atoi ( getc ( f1 ) );
                yref = atoi ( getc ( f2 ) );
                /* Initial values of v are unimportant */
                for ( t = 0, i = 1; i <= p; i++ )
                                t += N[i] * v[i] ;
                v[0] = yref - y + t ;
                for ( u = i = 0; i <= q; i++ )
                                u += D[i] * v[i] ;
                putc ( dtoa ( u ), f3 ) ;
                /* Shift v for next sample time */
                for ( i = p; i > 0; i-- )
                                v[i] = v[i-1] ;
                }
}
```

9. CONCLUSION

Discrete Polynomial Control Theory is in our opinion, a major step in the so-called counterrevolution against state space theory. It brings together the mathematical rigor of ''classical'' multivariable control theory, freed from the limitations of analog hardware, with the numerical tools developed in the last decade for reliable modeling of linear systems. Its power lies in its equal treatment of discrete and continuous, minimum- and nonminimum-phase, and low- and high-order systems, based on the impulse or step response. In the field of digital control, this is the first time we have extricated ourselves from the practical impossibility of state estimation. Even some recent works based on the time response representation of linear systems [7, 8] have fallen into the trap of state estimation and some have ended up requiring the solution of a nonlinear optimization problem *at each sample interval.* Discrete Polynomial Control is a powerful design method which is easily adaptable to real-time computer control, as each control action is mainly a matrix multiplication. Its complexity is independent of its order, except when impemented in systolic arrays, where the dimensions of the matrices affect the number of cells. It lends itself directly to sensitivity and robustness analysis, input/output constraints, disturbance rejection, dual sampling rates, nonsquare systems, and even time-varying systems.

REFERENCES

[1] Horowitz, I.M. and U. Shaked, ''Superiority of transfer function over state variable methods in linear time-invariant feedback system design'', IEEE Trans. on Automatic Control, vol. AC-20, no. 1 (1975)

[2] Kung, S.Y., ''A new identification and model reduction algorithm via singular value decomposition'', Proc. 12th Annual Asilomar Conference on Circuits, Systems, and Computers, p705 (1978)

[3] Kung, S.Y. and D.W. Lin, ''Recent progress in linear system model reduction via Hankel matrix approximation'', Proc. 1981 European Conference on Circuit Theory and Design, The Hague, p222 (1981)

[4] Genin, Y., ''An introduction to the model reduction problem with Hankel norm criterion'', *ibid*, p205

[5] Businger, P.A., ''Updating a singular value decomposition'', BIT, vol 10, p376 (1970)

[6] Wolovich, W.A., **Linear Multivariable Systems**, Springer-Verlag (1974)

[7] Richalet, J., A. Rault, J. L. Testud, and J. Papon, ''Model predictive heuristic control: Application to industrial processes'', Automatica, vol. 14, p413 (1978)

[8] Mehra, R.K., R. Rouhani, and L. Praly, ''New theoretical developments in multivariable predictive algorithmic control'', Proc. JACC, session FA9-B (1980).

ON THE EUCLIDEAN CONTROLLABILITY OF INFINITE DELAY SYSTEMS

A. S. C. SINHA
Purdue University at Indianapolis
P. O. Box 647, Indianapolis, Indiana 46223

ABSTRACT

In this paper, a question of Euclidean controllability of infinite delay systems with the coefficients A(t,s) and B(t,s) of bounded variation in s on $[t_0-h, t_0]$, is considered. On the basis of theorems due to Dauer and Gaul, a new sufficient condition for the Euclidean controllability is established.

1. INTRODUCTION

The basic problem of Euclidean controllability consists in deriving conditions on the systems coefficients, under which the solution of the control system can hit any point of the Euclidean space in finite time. Jacobs and Langenhop [1] treated autonomous linear neutral systems, while Sinha [2] considered nonlinear delay systems. These investigations are not as complete as the treatments of ordinary differential systems (see Ref. 3), or of delay systems in Ref. [7]. This paper is devoted to a study of Euclidean controllability on a bounded interval $I = [t_0, t_1]$ of the linear functional differential system of type,

$$\dot{x}(t) = \int_{t_0-h}^{t} d_s[A(t,s)]x(s) + \int_{t_0-h}^{t_0} d_s[\eta(t,s)]u(t+s) +$$

$$\int_{-\gamma}^{t_0} \mu(\theta)x(t+\theta)d\theta$$

$$x(t) = \psi(t) \quad \text{in } [t_0 - \gamma,\ t_0] \ ; \qquad (1)$$

and the integral is in Lebesgue-Stieltjes sense with respect to the variable "s".

Many authors have studied similar problems for ordinary differential systems (for references, see Ref. [8]). In a recent study, Dauer and Gahl in Ref. [9] considered nonlinear delay systems. Our approach here is motivated by Ref. [9]. We define the appropriate control and its corresponding solution by an integral equation. This equation is then solved to obtain Euclidean controllability of the infinite delay system (1) which extends previous results.

The following notational conventions will be used throughout the sequel. The symbol $L_2([t_0-h, t_1], E^m) = U$ denotes the class of square integrable functions $u: [t_0-h, t_1] \to E^m$ with values in the unit cube. Any $u \in U$ is called admissible.

Let $h > 0$ be given. For $u: [t_0-h, t_1] \to E^m$ and $t \in [t_0, t_1]$ we use the symbol u_t to denote the function on $[t_0-h, t_0]$ defined by $u_t(s) = u(t + s)$ for $s \in [t_0-h, t_0]$. The symbol $B([t_0-\gamma, t_1], E^n)$ with $h < \gamma$ (γ can be $+\infty$), denotes the Banach space of continuous functions $x: [t_0-\gamma, t_1] \to E^n$ with the usual sup topology, denoted by $||\cdot||\cdot$ BV [a,b] will represent the space of functions of bounded variation on $[a,b]$ with norm $||\cdot||$. The norm in E^p is denoted by $|\cdot|$. $|g| = \mathrm{Var}([a,b]; g) + |g(b)|$, where $\mathrm{Var}([a,b]; g)$ is the total variation of g on $[a,b]$. $\eta(t,s)$ is an n m matrix valued function which is measurable in $(t,s) \varepsilon E^2$. Also, $\eta(t,s)$ is of bounded variation in s on $[t_0-h, t_0]$ for each $t \varepsilon [t_0, t_1]$ with $\mathrm{Var}\, \eta(t,\cdot) \leq \nu(t)$ where $\nu(t)$ is a square integrable function from $[t_0, t_1] \rightarrow E$. $A(t,s) = 0$ for $s \geq t$; $A(t,s) = A(t, t_0-h)$, $s < t_0-h$; $A(t,s)$ is measurable, continuous from the left in s on $(-\infty,t)$; $s \rightarrow A(t,s)$ is of bounded variation on every finite s interval and $\mathrm{Var}([t_0-h, t]; A(t,\cdot)) \leq m(t)$ where m is locally integrable, and $\mu(\theta)$ is a nxn matrix function whose elements are Lebesque integrable on $(-\infty,0)$.

Definition 1.1.

Let

$$x_t(s) = x(t+s); \quad t_0-\gamma \leq s \leq t_0$$

$$u_t(s) = u(t+s). \quad t_0-h \leq s \leq t_0 \quad , \tag{2}$$

then the triple $z_t = (x(t), x_t, u_t) \varepsilon Z$, where $Z = E^n \times B([t_0-h, t_1], E^n) \times U$ is called a complete state of the system (1) at time t. By $x(t, z_{t_0}, u(\cdot))$ we denote a solution to (1) with initial complete state z_{t_0} and control $u(t)$, $t \varepsilon [t_0, t_1]$.

Definition 1.2

The system (1) is said to be Euclidean controllable on I if for every complete initial state z_{t_0}, and every vector $x_1 \varepsilon E^n$ there exists a square integrable function u: $[t_0, t_1] \rightarrow E^m$ such that $x(t_1, z_{t_0}, u(\cdot)) = x_1$.

2. PRELIMINARY RESULTS

The preliminary results are a consequence of Banks and Kent, Theorem 3.1 of Ref. [10]. We remark that $Y(s,t)$ is left continuous in its first argument. Let $\mu(\theta) \equiv 0$. Following results in [10], the unique continuous solution of (1) is given by

$$x\,(t; z_{t_0}, u(\cdot)) = Y(t_0,t)\, x_0 + \int_{t_0-h}^{t_0} d_s\Gamma(t,s)\psi(s) +$$

$$\int_{t_0}^{t} Y(s,t)\{\int_{t_0-h}^{t_0} d_r\, \eta(s,r)\, u(r+s)\}ds \tag{3}$$

and where

$$\Gamma(t,s) = \int_{t_0}^{t} Y(\alpha,t)\, A(\alpha,s)\, d\alpha \quad , \tag{4}$$

and Y is the unique solution of the system

$$Y(s,t) = I - \int_s^t Y(m,t)\, A(m,s)\, dm, \quad s \in [t_0,t]$$

$$Y(t,t) = I, \quad Y(s,t) = 0 \text{ for } s > t. \tag{5}$$

The last term in equation (3) contains values of $u(s)$ for $t < t_0$ as well as for $t \geqq t_0$. The values for $u(t)$ for $t < t_0$ enter into the definition of complete state. To separate them, the last term can be transformed by changing the order of integration where the unsymmetric Fubini theorem is used. From the Fubini theorem, we have

$$\int_{t_0}^{t} Y(s,t)\, \{\int_{t_0-h}^{t_0} d_r\, \eta(s,r)\, u(r+s)\}ds =$$

$$\int_{t_0-h}^{t_0} d_r\, \{\int_{t_0}^{t} Y(s,t)\, \eta(s,r)\, u(r+s)\, ds\} =$$

$$\int_{t_0-h}^{t_0} d_r\, \{\int_{t_0+r}^{t+r} Y(s-r,\, t)\, \eta(s-r,r)\, u(s)\, ds\} =$$

$$\int_{t_0-h}^{t_0} d_r\, \{\int_{t_0+r}^{t_0} \} + \int_{t_0-h}^{t_0} d_r\, \{\int_{t_0}^{t+r} \} \tag{6}$$

Since $r \leq 0$ so that $t+r \leq t$, and since $Y(s-r,t) \equiv 0$ for $s > t+r$, the integral in brackets can be extended to $s = t$, giving

$$\int_{t_0-h}^{t_0} d_r\, \{\int_{t_0}^{t} Y(s-r,t)\, \eta(s-r,r)\, u(s)\, ds\} =$$

$$\int_{t_0}^{t} \{\int_{t_0-h}^{t_0} Y(s-r,t)\, [d_r\, \eta(s-r,r)]\, u(s)\, ds\} \tag{7}$$

We now isolate the expression

$$\int_{t_0-h}^{t_0} Y(s-r,t)\, [d_r\, \eta(s-r,r)] \equiv \Phi(s,t) \tag{8}$$

and define it as the index of the control system (1).

We also observe that

$$\int_{t_0-h}^{t_0} d_r\, \{\int_{t_0+r}^{t_0} Y(s-r,t)\, \eta(s-r,r)\, u(s)\, ds\} =$$

$$\int_{t_0+r}^{t_0} \{\int_{t_0-h}^{t_0} Y(s-r,t)[d_r\, \eta(s-r,r)]\}\, u_{t_0}\, ds \tag{9}$$

With this solution of (1) with $\mu \equiv 0$, from equ. (3), is given by

$$x(t, z_{t_0}, u(\cdot)) = X(t, z_{t_0}, u_{t_0}) + \int_{t_0}^{t} \Phi(s,t)\, u(s)\, ds, \tag{10}$$

where

$$X(t, z_{t_0}, u_{t_0}) = Y(t_0,t)\, x_0 + \int_{t_0-h}^{\overset{-}{t_0}} d_s[\Gamma(t,s)]\psi(s) + \int_{t_0+r}^{t_0} \{\int_{t_0-h}^{t_0} Y(s-r,t)\, [d_r\eta(s-r,r)]\} u_{t_0}\, ds \tag{11}$$

Proposition 1. The system (1) with $\mu(\theta) \equiv 0$, is proper on $[t_0,t_1]$ if and only if the origin is an interior point of

$$R(t_1) = \int_{t_0}^{t_1} \Phi(s,t)\, u(s)\, ds$$

$$\Phi(s,t) \overset{\text{def.}}{=} \int_{t_0-h}^{t_0} Y(s-r, t)\, [d_r\, \eta(s-r, r)] \tag{12}$$

Proof. It is well-known that $R(t_1)$ is compact convex whenever $u(t)$ is such. Also, 0 in $R(t_1)$ for each $t \geq t_0$. $R(t_1)$ is a closed and convex subset of E^n, and there exists a point z_1 on the boundary of $R(t_1)$ such that

$$p^*(z - z_1) \leq 0 \tag{13}$$

for each z in $R(t_1)$ (see cor. 9-2 of [3]), where p is normal to the plane containing $R(t_1)$. If u_1 is the control corresponding to z_1, we have

$$p^* \int_{t_0}^{t_1} \Phi(s,t)\, u(s)\, ds \leq p^* \int_{t_0}^{t_1} \Phi(s,t)\, u_1(s)\, ds \tag{14}$$

for each u in a unit cube. The last inequality holds if and only if

$$p^* \int_{t_0}^{t_1} \Phi(s,t)\, u(s)\, ds \leq \int_{t_0}^{t_1} |p^* \Phi(s,t)\, u_1(s)| ds = z_1 \leq \int_{t_0}^{t_1} |p^* \Phi(s,t)| ds \tag{15}$$

where let $u_1(s) \equiv \mathrm{sgn}[p^* \Phi(s,t)]$. Since we always have 0 in $R(t_1)$ if 0 is not in the interior of $R(t_1)$, then 0 is in the boundary.

$$0 = \int_{t_0}^{t_1} |p^* \Phi(s,t)| ds \tag{16}$$

which implies that

$$p^* \Phi(s,t) = 0, \text{ a.e. } s \text{ in } [t_0,t_1]. \tag{17}$$

Thus, from the definition of proper, the proof of the proposition is complete.

Proposition 2. The system (1) with $\mu(\theta) = 0$ is Euclidean controllable on $[t_0,t_1]$ if, and only if,

$$W(t_0,t_1) = \int_{t_0}^{t_1} \Phi(s,t_1)\, \Phi^T(s,t_1)\, ds \tag{18}$$

$$\Phi(s,t) = \int_{t_0-h}^{t_0} Y(s-r,\, t)[d_r\, \eta(s-r,\, r)] \tag{19}$$

is nonsingular. The matrix Y is defined by equ. (5).

The proof of the Proposition 2 follows exactly as the Proposition in [9].

Theorem 1. Let

$$\alpha(t) = \sup_{-\infty < \theta < 0} \frac{|\mu(\theta-t)|}{|\mu(t)|} \tag{20}$$

where α is continuous and $|\alpha(t)| < 1$. If the conditions of Proposition 1 and 2 hold, then the system (1) is controllable on $I = [t_0,t_1]$.

The proof for the paper is omitted due to lack of space.

REFERENCES

1. Jacobs, M. Q., and Langenhop, C. E., Criteria for Function Space Controllability of Linear Neutral Systems, SIAM Journal on Control and Optimization, Vol. 14, pp. 1009-1048, 1976.

2. Sinha, A. S. C., Controllability of Nonlinear Delay Systems, Int. J Control, Vol. 43, pp. 1305-1315, 1986.

3. Hermes, H., and LaSalle, J. P., Functional Analysis and Time Optimal Control, Academic Press, New York, New York, 1969.

4. Olbrot, A. W., On Controllability of Linear Systems with Time Delays in Control. IEEE Transactions on Automatic Control, Vol. AC-17, pp. 664-666, 1969.

5. Manitius, A., and Olbrot, A. W., Controllability Conditions for Systems with Delays in State and Control, Archiwum Automatykii Telemechaniki, Vol. 17, pp. 119-131, 1972.

6. Manitius, A., On the Controllability Conditions for Systems with Distributed Delays in State Control, Archiwum Automatykii Telemechaniki, Vol. 17, pp. 363-376, 1972.

7. Sebakhy, O., and Bayoumi, N. M., Controllability of Linear Time-varying Systems with Delay in Control. International Journal of Control, Vol. 17, pp. 127-135, 1973.

8. Dauer, J. P., Nonlinear Perturbations of Quasi-linear Control Systems, Journal of Mathematical Analysis and Applications, Vol. 54, pp. 717-725, 1976.

9. Dauer, J. P., and Gahl, R. D., Controllability of Nonlinear Delay Systems, Journal of Optimization Theory and Applications, Vol. 21, pp. 59-70, 1977.

10. Banks, H. T., and Kent, G. A., Control of Functional Differential Equations of Retarded and Neutral Type to Target Sets in Function Space, SIAM Journal on Control, Vol. 10, pp. 567-593, 1972.

FAULT TOLERANT CONTROLLER DESIGN USING MULTIPLE FEEDBACK LOOPS

J. GLOWER and Ü. ÖZGÜNER
Department of Electrical Engineering
The Ohio State University
2015 Neil Avenue
Columbus, OH 43210

INTRODUCTION

Two approaches are utilized to design fault tolerant control systems: either a fixed control law is adopted so that the system is guaranteed to be stable even after a fault occurs, or a routine that detects when a fault occurs, identifies the component that failed, and then presumably reconfigures the control accordingly, is used. In the present paper, a simple procedure for calculating two sets of feedback gains is given such that closed loop stability is guaranteed even after one set of feedback gains is lost. Furthermore, the degree of suboptimality of the failed system may be predefined.

DEVELOPMENTS

Consider a linear, time-invariant system

$$\dot{x} = Ax + Bu \tag{1}$$

$$u = Kx \tag{2}$$

where $[A, B]$ is a controllable pair. The problem to be solved is defined as follows: "Find two sets of feedback gains, K_0 and K_1 , such that the closed loop system remains stable if either or both are fed back to the system". The following algorithm is proposed:

Algorithm:

Step 1: Pick a positive definite matrix Q_0, such that:

$$(A + Q_0) > 0 \tag{3}$$

Define K_0 to be

$$K_0 = -B^T P_0^{-1} \tag{4}$$

where P_0 is the solution of the Lyapunov equation

$$(A + Q_0)P_0 + P_0(A + Q_0)^T = 2BB^T \tag{5}$$

Step 2: Pick a positive definite matrix Q_1 such that

$$(A + Q_1) > 0 \quad (Q_1 - BK_0) > 0 \tag{6}$$

Step 3: Pick K_1 to be

$$K_1 = -B^T P_1^{-1} \tag{7}$$

where P_1 satisfies eqn. (5) with the subscript 1.

It is claimed that the feedback control

$$K = K_0 + K_1 \tag{8}$$

will stabilize the system (2) and the closed loop system will remain stable if either one of the feedback gains is set to zero. Proof of this claim can be done based on the results in Ref.[1]. Due to space restrictions, details will be omitted.

Often, one is interested not only in the stability of a system after failure, but also in the ensuing suboptimality. Consider the infinite time quadratic regulator specified by the weighting [Q,I]. The suboptimality of a stabilizing feedback gain K is defined to be:

$$\eta = trace(P_{nominal})/trace(P_n) \tag{9}$$

where P is the solution of the Lyapunov equation

$$(A + BK)^T P + P(A + BK) = Q \tag{10}$$

and Q is chosen such that the cost is minimum for the nominal case. Assume now that in Step 2 of the algorithm above we select $Q_1 = \alpha Q_0$. It is now claimed that as Q tends to infinity the performance of the system after failure of K_0 or K_1 is $\alpha/(1+\alpha)$ or $1/(1+\alpha)$, respectively. Again, the proof is omitted due to space restrictions.

References

[1] Ü. Özgüner and P.E. Sarachik, "Stabilizing Feedback Gains for Priority-Ordered Multichannel Linear Systems," IEEE Trans. Auto. Contr. AC-25. No. 1. 104-106, Feb. 1980.

STATE OBSERVATION OF NONLINEAR SYSTEMS: A COMPARATIVE STUDY

B.L. WALCOTT and S.H. ŻAK
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907

Abstract: This paper contains a comparative study of four techniques for observing the states of nonlinear systems. The first technique examined is inspired by Bestle and Zeitz and Krener and Respondek. In this method, a nonlinear transformation is found which brings the system into a canonical form, from where observer design is facilitated. The second technique is attributable to Thau. In this method, the error between the system's true state and the output of the observer is shown to be asymptotically convergent to zero provided that an additional assumption is valid. The third technique is due to Baumann and Rugh. In this method, extended or pseudolinearization of the error differential equation about a family of equilibrium points results in an observer design such that the eigenvalues of the linearized error equation are locally invariant. Finally, techniques from variable structure systems are utilized to design an observer which yields an exponentially decaying error like Thau's observer, but, unlike Thau's, does not incorporate the nonlinearities of the system into the observer design. An example illustrating the performance of the four techniques is included.

1. INTRODUCTION

State feedback control of nonlinear systems is a topic of recent interest in the literature (e.g., [1-3]). Complete accessibility of the states is a commonly invoked assumption in such control schemes. Since, in general, this is not a valid assumption, design of an observer which generates an estimate which converges at least asymptotically to the actual states is a problem which has received much attention (e.g., [4-6]).

In this paper, we will apply Lie Algebraic techniques to the problem of nonlinear system state observation as suggested by Bestle and Zeitz [7] and Krener and Respondek [8]. Thau's [9] approach to nonlinear observation is presented followed by the extended linearization technique of Baumann and Rugh [10]. Finally, an observer design technique utilizing the theory of variable structure systems (VSS) is examined [11]. A computer simulation example illustrating the performance of each of these four approaches precedes the concluding section.

2. TRANSFORMATION OF NONLINEAR SYSTEMS INTO OBSERVER FORM

2.1 Single Output Case

Consider the following class of time invariant, nonlinear systems described by the following equations

$$\dot{x} = f(x) \qquad x(t_0) = x_0 \tag{1}$$

$$y = h(x) \tag{2}$$

where $x \in \mathbb{R}^n$ and $y \in \mathbb{R}$. We will assume that $f(\cdot)$ is a C^∞ vector field on $\mathbb{R}^n$ and $h(\cdot)$ is a C^∞ function. We desire to find a one-to-one, C^∞, nonlinear transformation, $T:\mathbb{R}^n \to \mathbb{R}^n$ where

$$x = T(x^*) \tag{3}$$

such that system (1) may be transformed into observer form as defined below

$$\dot{x}^* = \begin{bmatrix} 0 & \cdots & & 0 \\ 1 & & & \cdot \\ & \cdot & & \cdot \\ & & \cdot & \cdot \\ & & \cdot & \\ 0 & & 1 & 0 \end{bmatrix} x^* - \begin{bmatrix} f_0^*(x_n^*) \\ f_1^*(x_n^*) \\ \cdot \\ \cdot \\ \cdot \\ f_{n-1}^*(x_n^*) \end{bmatrix} \triangleq f^*(x^*) \tag{4}$$

and

$$y = [0,\ldots,0,1]x^*.$$

In the ensuing analysis we will follow the arguments of Bestle and Zeitz [7] yet utilizing Lie Algebraic notation as proposed in Krener and Respondek [8] or Hunt et al. [3].

Taking the derivative of (3) with respect to time yields

$$\dot{x} = \frac{\partial T}{\partial x^*} f^*(x^*) \tag{5}$$

where

$$\frac{\partial T}{\partial x^*} = [\frac{\partial T}{\partial x_1^*}, \ldots, \frac{\partial T}{\partial x_n^*}]. \tag{6}$$

Equating the right hand sides of (1) and (5) cedes

$$f(x) = f(T(x^*)) = \frac{\partial T}{\partial x^*} f^*(x^*). \tag{7}$$

Taking the partial derivative of (7) with respect to x_k^* yields

$$\frac{\partial f(x)}{\partial x_k^*} = \frac{\partial[\frac{\partial T}{\partial x^*}]}{\partial x_k^*} f^*(x^*) + \frac{\partial T}{\partial x^*}\frac{\partial f^*(x^*)}{\partial x_k^*}. \tag{8}$$

Note that

$$\frac{\partial f(T(x^*))}{\partial x_k^*} = \frac{\partial f(x)}{\partial x} \cdot \frac{\partial T(x^*)}{\partial x_k^*}. \tag{9}$$

Similarly,

$$\frac{\partial[\frac{\partial T(x^*)}{\partial x^*}]}{\partial x_k^*} f^*(x^*) = \frac{\partial(\frac{\partial T}{\partial x_k^*})}{\partial x}\frac{\partial T(x^*)}{\partial x^*} f^*(x^*) = \frac{\partial(\frac{\partial T}{\partial x_k^*})}{\partial x} f(x). \tag{10}$$

From the structure of $f^*(x^*)$ (given in (4)) we find that

$$\frac{\partial T}{\partial x^*}\frac{\partial f^*(x^*)}{\partial x_k^*} = \begin{cases} \dfrac{\partial T}{\partial x_{k+1}^*} & k = 1,...,n-1 \\ -\dfrac{\partial T}{\partial x^*}\dfrac{\partial f^*}{\partial x_n^*}. & \end{cases} \tag{11}$$

Upon substitution of (9), (10) and (11) into (8) we obtain

$$\frac{\partial T}{\partial x_{k+1}^*} = \frac{\partial f}{\partial x}\frac{\partial T}{\partial x_k^*} - \frac{\partial(\frac{\partial T}{\partial x_k^*})}{\partial x} f(x) \qquad \text{for } k = 1,...n-1, \tag{12}$$

and

$$\frac{\partial T}{\partial x^*}\frac{\partial f^*(x^*)}{\partial x_n} = \frac{\partial f}{\partial x}\frac{\partial T}{\partial x_n^*} - \frac{\partial(\frac{\partial T}{\partial x_n^*})}{\partial x} f(x). \tag{13}$$

To obtain further simplification, we can employ Lie derivative notation. Observe that equation (12) can be rewritten in the following manner

$$\frac{\partial T}{\partial x^{*}{}_{k+1}} = [f, \frac{\partial T}{\partial x_k^*}] = (ad^1 f, \frac{\partial T}{\partial x_k^*}) \qquad k = 1,...,n-1, \tag{14}$$

and

$$-\frac{\partial T}{\partial x_n^*}\frac{\partial f^*}{\partial x_n^*} = (ad^1 f, \frac{\partial T}{\partial x_n^*}) = [f, \frac{\partial T}{\partial x_n^*}]. \tag{15}$$

Note that

$$\frac{\partial T}{\partial x_2^*} = (ad^1 f, \frac{\partial T}{\partial x_1^*}), \tag{16}$$

and

$$\frac{\partial T}{\partial x_3^*} = (ad^1 f, \frac{\partial T}{\partial x_2^*}) = (ad^2 f, \frac{\partial T}{\partial x_1^*}). \tag{17}$$

Therefore, in general

$$\frac{\partial T}{\partial x_k^*} = (ad^{k-1} f, \frac{\partial T}{\partial x_1^*}) \qquad k = 1,...,n. \tag{18}$$

It is now possible to express all the columns of $\frac{\partial T}{\partial x^*}$ in terms of the single "starting vector" [7], $\frac{\partial T}{\partial x_1^*}$, as follows

$$\frac{\partial T}{\partial x^*} = [(ad^0 f, \frac{\partial T}{\partial x_1^*}),(ad^1 f, \frac{\partial T}{\partial x_1^*}),...,(ad^{n-1} f, \frac{\partial T}{\partial x_1^*})]. \tag{19}$$

Next, we will employ the equation,

$$y = h(x) = x_n^* \tag{20}$$

to obtain an expression for the starting vector, $\frac{\partial T}{\partial x_1^*}$. Taking the partial derivative of (20) with respect to x^* yields the following

$$\frac{\partial h(x)}{\partial x}\frac{\partial T}{\partial x^*} = [0,0,\ldots,0,1]. \tag{21}$$

Note that the first component of equation (21) may be written as

$$\frac{\partial h}{\partial x}\frac{\partial T}{\partial x_1^*} = <dh,\frac{\partial T}{\partial x_1^*}> = L_f^0(dh)(x)\frac{\partial T}{\partial x_1^*} = 0. \tag{22}$$

Similarly, we may use Leibnitz's formula [3] to simplify the second element of (21):

$$\frac{\partial h}{\partial x}\frac{\partial T}{\partial x_2^*} = <dh,[f,\frac{\partial T}{\partial x_1^*}]> = <d<dh,f>,\frac{\partial T}{\partial x_1^*}> - <d<dh,\frac{\partial T}{\partial x_1^*}>,f> = L_f^1(dh)(x)\frac{\partial T}{\partial x_1^*} = 0. \tag{23}$$

By repeated application of Leibnitz's formula and use of (19) we may obtain the following matrix equation involving the starting vector, $\frac{\partial T}{\partial x_1^*}$

$$\begin{bmatrix} L_f^0(dh)(x) \\ L_f^1(dh)(x) \\ \vdots \\ L_f^{n-1}(dh)(x) \end{bmatrix} \frac{\partial T}{\partial x_1^*} \triangleq O\,\frac{\partial T}{\partial x_1^*} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \tag{24}$$

where the matrix O is termed the observability matrix of the system defined by (1), (2). Thus, the starting vector, $\frac{\partial T}{\partial x_1^*}$ is equal to the last column of O^{-1}.

2.2 Multi-Output Case

For the multi-output case see Walcott, et. al. [12].

3. OBSERVER DESIGN

The advantage of employing the preceding technique to put the system into observer form is that observer design is greatly simplified in the new coordinate system.

For the single output case, consider the observer governed by the equation

$$\dot{\hat{x}}^* = \begin{bmatrix} 0 & \cdot & \cdot & \cdot & 0 \\ 1 & & & & \cdot \\ & \cdot & & & \cdot \\ & & \cdot & & \cdot \\ & & & \cdot & \\ 0 & & & 1 & 0 \end{bmatrix} \hat{x}^* - \begin{bmatrix} f_0^*(x_n^*) \\ f_1^*(x_n^*) \\ \cdot \\ \cdot \\ \cdot \\ f_{n-1}^*(x_n^*) \end{bmatrix} - K(\hat{y} - y), \tag{25}$$

where $\hat{y} = [0\ldots01]\hat{x}^*$ and $K = [k_0 \cdots k_{n-1}]^T$. Define the error as $e = \hat{x}^* - x^*$. Hence, the error obeys a homogeneous differential equation with characteristic polynomial of

$$p(s) = k_0 + k_1 s + \cdots + k_{n-1}s^{n-1} + s^n \tag{26}$$

Thus, we may easily assign the spectrum of (26) via an appropriate selection of K.

Based on the results of [3] it can be shown that a diffeomorphism, $T(\cdot)$, which transforms system (1), (2) into the form given by (4) preserves the eigenvalues of the linearized error equation in the original coordinate system.

4. OBSERVER DESIGN BY EXTENDED LINEARIZATION

Baumann and Rugh [10] offer an observer design technique for nonlinear systems which yields constant eigenvalues of the error differential equation when linearized about any fixed equilibrium points.

Consider the system described by

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= h(x) \end{aligned} \tag{27}$$

where f and g are analytic and

$$f(\cdot,\cdot):\mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n \quad \text{with } f(0,0) = 0 \tag{28}$$

$$h(\cdot):\mathbb{R}^n \rightarrow \mathbb{R}^p \quad \text{with } h(0) = 0. \tag{29}$$

We will assume that (27) has a family of constant operating points denoted by $u = \epsilon$ and $x = x_\epsilon$. That is,

$$f(x_\epsilon,\epsilon) = 0. \tag{30}$$

Let the observer estimate, $\hat{x}$, obey

$$\dot{\hat{x}} = f(\hat{x},u) + g(y) - g(\hat{y}) \tag{31}$$

where $\hat{y} \triangleq h(\hat{x})$ and $g(\cdot):\mathbb{R}^p \to \mathbb{R}^n$ is analytic with $g(0)=0$. If we denote $e(t) = x - \hat{x}$, then

$$\dot{e} = f(x,u) - f(x-e,u) - g(y) + g(\hat{y}) \tag{32}$$

Note that $u = \epsilon$, $x = x_\epsilon$, $e = 0$ and $\hat{y}_\epsilon = y_\epsilon = h(x_\epsilon)$ is an equilibrium point of (32). Expanding equation (32) about this equilibrium point and neglecting higher order terms yields

$$\dot{e} = [D_1f(x_\epsilon,\epsilon) - Dg(y_\epsilon)Dh(x_\epsilon)]e \tag{33}$$

where $Dh(x) \triangleq \partial\frac{h(x)}{\partial x}$ and $D_1f(x,u) \triangleq \frac{\partial f(x,u)}{\partial x}$. Let us impose the following constraints:

1. $D_1f(0,0)^{-1}$ exists
2. $(D_1f(0,0),Dh(0))$ is completely observable
3. $\left.\frac{\partial y_\epsilon}{\partial \epsilon}\right|_{\epsilon=0} = \left.Dy_\epsilon\right|_{\epsilon=0} = Dh(0)Dx_\epsilon(0) \neq 0$

where $D_2f(x,u) \triangleq \frac{\partial f(x,u)}{\partial u}$.

The aim of this observer design is to find g which is analytic such that the eigenvalues of (33) are locally invariant with respect to ϵ.

Note that the eigenvalues of (33) are the same as the eigenvalues of

$$[D_1f(x_\epsilon,\epsilon)^T - Dh(x_\epsilon)^T Dg(y_\epsilon)^T]. \tag{34}$$

Recall from constraint #2, that the pair $[D_1f(0,0),Dh(0)]$ is completely observable. Therefore, the pair $[D_1f(0,0)^T,Dh(0)^T]$ is completely controllable. This implies that we may use Ackermann's synthesis formula [13] to find $C(\epsilon)$ where $C(\cdot):\mathbb{R} \to \mathbb{R}^{p\times n}$ such that

$$D_1f(x_\epsilon,\epsilon)^T - Dh(x_\epsilon)^T C(\epsilon)$$

has a prespecified spectrum which is independent of ϵ.

Thus, if we can find an analytic g such that the following partial differential equation is satisfied

$$Dg(y_\epsilon)^T = C(\epsilon)$$

the linearized observer error equation (33) will have eigenvalues independent of ϵ. Let us define

$$C^T(\epsilon) = [c_1(\epsilon),\ldots,c_p(\epsilon)]$$

where $c_i(\epsilon)$ $(i=1,\ldots,p)$ are $(n\times 1)$ vectors.

Without loss of generality we may assume that the first element of the p vector, Dy_ϵ in constraint #3 is nonzero (i.e $Dy_{\epsilon_1} \neq 0$). Thus, by the inverse function theorem we can find $y_{\epsilon_1}^{-1}$ such that

$$y_{\epsilon_1}^{-1}(y_{\epsilon_1}) = \epsilon.$$

Hence, we can express y_ϵ as an explicit function of ϵ:

$$y_\epsilon = y(\epsilon).$$

Let $\hat{g}(\epsilon) = g(y(\epsilon))$ which implies that

$$D\hat{g}(\epsilon) = \frac{\partial g}{\partial y(\epsilon)}\cdot\frac{\partial y(\epsilon)}{\partial \epsilon} = C^T(\epsilon)Dy_\epsilon. \tag{35}$$

Note that we can now find Dy_ϵ and, therefore, by integrating (35) we can find $\hat{g}(\epsilon)$.

We claim that

$$g(y) = \hat{g}(y_{\epsilon_1}^{-1}(y_1)) + \sum_{j=2}^{p} c_j(y_{\epsilon_1}^{-1}(y_1))[y_j - y_{\epsilon_j}(y_{\epsilon_1}^{-1}(y_1))]. \tag{36}$$

For substantiation of this claim, see [12].

5. THAU'S OBSERVER

Another approach to observing the states of a nonlinear system was offered by Thau [9] whose results were later generalized by Kou et al. [5]. This method does not include a systematic technique for construction of the observer, but rather gives a sufficient condition for asymptotic stability of the origin of the error differential equation.

Consider the nonlinear system governed by

$$\begin{aligned} x &= Ax + f(x) + Bu \\ y &= Cx \end{aligned} \tag{37}$$

where $f(\cdot):\mathbb{R}^n \rightarrow \mathbb{R}^n$ is continuous and $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{p \times n}$. The nonlinear function $f(x)$ may also contain linear terms in x. We assume that the pair (A,C) is completely observable. Therefore, we can find $K \in \mathbb{R}^{n \times p}$ such that the eigenvalues of $A_0 = A - KC$ are in the open left-half plane (LHP).

Let $\hat{x}$, the estimate of the true state, obey the equation

$$\dot{\hat{x}} = A_0\hat{x} + f(\hat{x}) + Ky + Bu, \tag{38}$$

and let $e(t) = \hat{x}(t) - x(t)$. Thus, e(t) satisfies the following differential equation

$$\dot{e}(t) = A_0 e(t) + f(\hat{x}) - f(x) = A_0 e(t) + f(x + e) - f(x). \tag{39}$$

Since the spectrum of A_0 is contained in the LHP, then for any given positive definite $Q \in \mathbb{R}^{n \times n}$ there exists a unique positive definite $P \in \mathbb{R}^{n \times n}$ such that

$$A_0^T P + PA_0 = -2Q$$

(see [14]). Next, consider the following positive definite Lyapunov function candidate

$$V(e) = e^T Pe.$$

The derivative of V(e) evaluated along the solution of the error differential equation (39) is given by

$$\dot{V}(e) = \dot{e}^T Pe + e^T P\dot{e} = -2e^T Qe + 2e^T P[f(x + e) - f(x)]. \tag{40}$$

We impose the additional constraint that the function $f(\cdot)$ is locally Lipschitz about the origin, that is there exists a positive constant L such that

$$\|f(x_1) - f(x_2)\| \leq L\|x_1 - x_2\|$$

for all x_1, x_2 in some region R containing the origin. Therefore, if e(0) is contained in R, then the following inequalities are valid

$$\dot{V}(e) \leq -2e^T Qe + 2L\|Pe\|\|e\| \leq (-2a + 2L)\|Pe\|\|e\|$$

where a is the minimum eigenvalue of Q and $\|P\|$ is the maximum eigenvalue of P. Hence, if

$$\frac{a}{\|P\|} > L \tag{41}$$

then $e(t)=0$ is an asymptotically stable equilibrium point of equation (39).

6. OBSERVER DESIGN BASED ON VSS TECHNIQUES

Consider the class of systems described by

$$\begin{aligned} \dot{x}(t) &= Ax(t) + f(t,x) + B(u(t) + v(t)) \\ y(t) &= Cx(t) \end{aligned} \tag{42}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$ and $y \in \mathbb{R}^p$. The function $f(\cdot)$ can be interpreted as the nonlinearities and/or uncertainties in the plant although it is permissible for $f(\cdot)$ to contain terms which are linear in the state while $v(\cdot)$ can be construed as input disturbances. For existence purposes, we require that f(t,x) be continuous in x. In addition, let us impose the following three restrictions on system (42).

1. The pair [A,C] is detectable.

Let K be a matrix of consistent dimensions such that the spectrum of $A_0 \triangleq A - KC$ is completely contained in the open left half plane.

2. There exists a positive definite matrix $Q \in \mathbb{R}^{m \times n}$ and functions h and w where $h(\cdot,\cdot):\mathbb{R}^1_+ \times \mathbb{R}^n \rightarrow \mathbb{R}^p$ and $w(\cdot):\mathbb{R}^1_+ \rightarrow \mathbb{R}^p$ such that

$$f(t,x) = P^{-1}C^T h(t,x) \quad \text{and} \quad Bv(t) = P^{-1}C^T w(t)$$

where P is the unique, positive definite solution to the Lyapunov equation

$$PA_0 + A_0^T P = -Q$$

Let $\varsigma \triangleq h(t,x) + w(t)$.

3. There exists a positive function ρ where $\rho(\cdot):\mathbb{R}^1_+ \rightarrow \mathbb{R}^1_+$ such that

$$\|\varsigma(t,x)\| \leq \rho(t) \qquad \forall x \in \mathbb{R}^n \text{ and } t \in \mathbb{R}^1_+.$$

Finally, let the error difference between the observer estimate, $\hat{x}$, and the true state be denoted as

$$e = \hat{x} - x$$

Note that $Ce = C\hat{x} - y$. Under these three restrictions, consider the observer governed by the following differential equation

$$\dot{\hat{x}} = A_0\hat{x} + S(\hat{x},y) + Ky + Bu \tag{43}$$

where

$$S(\hat{x},y) = \begin{cases} \dfrac{-P^{-1}C^TCe}{||Ce||}\rho & \forall e \notin N \\ 0 & \forall e \in N \end{cases}$$

and $N = \{e: Ce = 0\}$. The error obeys the differential equation

$$\dot{e} = A_0e - \frac{P^{-1}C^TCe}{||Ce||}\rho - P^{-1}C^T\varsigma.$$

It can be shown that not only is e(t) globally uniformly asymptotically stable to the origin, but e^TPe, which can be viewed as the distance of e(0) from the origin, decays at least exponentially [11]. That is,

$$e(t)^TPe(t) \leq e(0)^TPe(0)\exp[-\eta(t-t_0)] \quad \forall t \geq t_0$$

where η is the minimum eigenvalue of $P^{-1}Q$. In the case when $v(t) \equiv w(t)$ and $m=p$, the second condition of assumption #2 takes the form

$$PB = C^T.$$

For a system theoretic interpretation of this condition see Steinberg and Corless [15].

7. COMPUTER SIMULATION EXAMPLE

As a final comparative example for the four observer techniques reviewed in this paper consider a system modeled by the following relationships

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ -\sin x_1\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}u = f(x) + Bu \tag{44}$$

$$y = x_1 + x_2 = h(x).$$

7.1 The Lie Algebraic Method

Since (55) is linear in the control, we may consider the control to be zero without loss of generality. If the control were nonzero, we may easily add Bu to observer dynamics in the original coordinates.

The starting vector, $\frac{\partial T}{\partial x_1^*}$, is the last column of O^{-1} and evaluates to

$$\frac{\partial T}{\partial x_1^*} = \begin{bmatrix} \dfrac{-1}{\cos x_1 + 1} \\ \dfrac{1}{\cos x_1 + 1} \end{bmatrix}.$$

The Jacobian matrix of T evaluates to

$$\frac{\partial T}{\partial x^*} = [(ad^0f, \frac{\partial T}{\partial x_1^*}),(ad^1f, \frac{\partial T}{\partial x_1^*})] = \begin{bmatrix} \dfrac{-1}{\cos x_1 + 1} & \dfrac{1 + \cos x_1 - x_2\sin x_1}{(\cos x_1 + 1)^2} \\ \dfrac{1}{\cos x_1 + 1} & \dfrac{\cos^2 x_1 + \cos x_1 - x_2 \sin x_1}{(\cos x_1 + 1)^2} \end{bmatrix}.$$

The above partial differential equation cannot be easily solved for x^*. Rather than abandon the Lie Algebraic technique at this point, we will employ an alternative approach as presented in Bestle and Zeitz (1983). This method is based upon linearizing observer equation (25) about the estimate $\hat{x}_n^*$. For the system

$$\dot{x} = f(x) \qquad x(t_0) = x_0$$

$$y = h(x)$$

the alternative observer design is given by the equation

$$\dot{\hat{x}} = f(\hat{x}) + g(\hat{x})[y - h(\hat{x})] \qquad \hat{x}(t_0) = \hat{x}_0$$

where

$$g(\hat{x}) = \frac{\partial T}{\partial \hat{x}^*}[k - \frac{\partial f^*}{\partial \hat{x}_n}].$$

The elements of the vector, $k = [k_0,...,k_{n-1}]^T$, are the coefficients of the desired characteristic polynomial (27). Under the assumption that

$$\left.\frac{\partial f^*(x_n^*)}{\partial x_n^*}\right|_{\hat{x}_n^*} \equiv \frac{\partial f^*(\hat{x}_n^*)}{\partial \hat{x}_n^*}$$

we may find $\frac{\partial T}{\partial \hat{x}^*}\frac{\partial f^*}{\partial \hat{x}_n^*}$ from equation (13).

For our system, if we choose the desired characteristic polynomial,

$$p(s) = k_0 + k_1 s + s^2 = 1 + 2s + s^2$$

then the alternative observer design is given by the relations

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \hat{x}_2 \\ -\sin\hat{x}_1 \end{bmatrix} \begin{bmatrix} g_1(\hat{x}) \\ g_2(\hat{x}) \end{bmatrix} \tag{45}$$

where

$$g_1(\hat{x}) = (\cos\hat{x}_1 + 1)^{-3}[\cos^3\hat{x}_1(\hat{x}_2^2 + 3)\cos^2\hat{x}_1 + (3 + \hat{x}_2^2 - 2\hat{x}_2\sin\hat{x}_1 - \sin^2\hat{x}_1)\cos\hat{x}_1 + (2\hat{x}_2^2 - 1)\sin^2\hat{x}_1 - 2\hat{x}_2^2\sin\hat{x}_1 + 1],$$

and

$$g_2(\hat{x}) = (\cos\hat{x}_1 + 1)^{-3}[\cos^3\hat{x}_1 + (3 + \hat{x}_2^2 + \hat{x}_2\sin\hat{x}_1)\cos^2\hat{x}_1 + (3 + 4\hat{x}_2\sin\hat{x}_1 + \sin^2\hat{x}_1 + \hat{x}_2^2)\cos\hat{x}_1 + (2\hat{x}_2^2 - 1)\sin^2\hat{x}_1 + 3\hat{x}_2\sin\hat{x}_1 + 1].$$

7.2 The Method of Extended Linearization

Equation (55) has constant operating points for $u = \epsilon$ and $x_{1_\epsilon} = \sin^{-1}(\epsilon)$, $x_{2_\epsilon} = 0$. In the method of extended linearization, the observer obeys the following equation

$$\dot{\hat{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\hat{x} + \begin{bmatrix} 0 \\ -\sin\hat{x}_1 + u \end{bmatrix} + g(y) - g(\hat{y}).$$

Thus, the error, $x - \hat{x}$, is governed by

$$\dot{e} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} e - \begin{bmatrix} 0 \\ \sin x_1 - \sin(x_1 + e) \end{bmatrix} + g(y) - g(\hat{y}). \tag{46}$$

Recall that it is our aim to find $g(\cdot)$, analytic, such that equation (46) possesses locally invariant eigenvalues when linearized about the constant operating points

$$u = \epsilon;\ x_{1_\epsilon} = \sin^{-1}(\epsilon), x_{2_\epsilon} = 0;\ e = 0;\ y_\epsilon = \hat{y}_\epsilon = \sin^{-1}(\epsilon).$$

Upon performing this linearization, we obtain

$$\dot{e} = \left(\begin{bmatrix} 0 & 1 \\ \cos(\sin^{-1}(\epsilon)) & 0 \end{bmatrix} - Dg(y_\epsilon)\begin{bmatrix} 1 & 0 \end{bmatrix} \right) e.$$

Let us define the following entities

$$A = \begin{bmatrix} 0 & -\cos(\sin^{-1}(\epsilon)) \\ 1 & 0 \end{bmatrix} \qquad b = Dh(x_\epsilon)^T = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

We may assign the spectrum of A-bk via the use of Ackermann's synthesis formula,

$$k = e_1 P(A)$$

where e_1 is the last row of the inverse of the controllability matrix defined by the pair [A,b] and $P(\cdot)$ is the desired characteristic polynomial of the linearized error equation (46). Choosing the eigenvalues of the linearized error equation to be {-1,-1} yields the ensuing two equations

$$e_1 = \begin{bmatrix} \dfrac{-1}{\cos(\sin^{-1}(\epsilon)) + 1} & \dfrac{1}{\cos(\sin^{-1}(\epsilon)) + 1} \end{bmatrix}$$

and

$$P(A) = A^2 + 2A + I = \begin{bmatrix} 1-\cos(\sin^{-1}(\epsilon)) & -2\cos(\sin^{-1}(\epsilon)) \\ 2 & 1-\cos(\sin^{-1}(\epsilon)) \end{bmatrix}.$$

Therefore,

$$Dg(y_\epsilon) = k^T \triangleq C(\epsilon)^T = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Note that

$$Dy_\epsilon = \frac{1}{\sqrt{1 - \epsilon^2}}$$

and

$$D\hat{g}(\epsilon) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \frac{1}{\sqrt{1-\epsilon^2}}. \tag{47}$$

Integrating (47) with respect to ϵ and employing equation (36) produces the desired $g(\cdot)$

$$g(y) = \begin{bmatrix} y \\ y \end{bmatrix}$$

Hence, the resulting observer is governed by

$$\dot{\hat{x}} = \begin{bmatrix} -1 & -1 \\ -1 & 0 \end{bmatrix} \hat{x} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} y - \begin{bmatrix} 0 \\ -\sin\hat{x}_1 + u \end{bmatrix}. \tag{48}$$

Thau's Technique

To invoke the method of Thau to our example, we will first rewrite the system dynamics as follows

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ -\sin x_1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u = Ax + f(x) + Bu \tag{49}$$

$$y = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = Cx$$

Again, without loss of generality, we may assume $u = 0$. Note that the pair $[A,C]$ is completely observable. Additionally, we will assign the spectrum $\{-1, -1\}$ to $A_0 = A-KC$. Thus, the Thau observer is given by

$$\dot{\hat{x}} = \begin{bmatrix} -1 & 0 \\ -1 & -1 \end{bmatrix} \hat{x} + \begin{bmatrix} 0 \\ -\sin\hat{x}_1 \end{bmatrix} + \begin{bmatrix} 1 & 1 \end{bmatrix} y. \tag{50}$$

Note that the Thau observer is identical to the observer obtained by the method of extended linearization. Unfortunately, sufficient condition (41) which guarantees that the observer estimate, $\hat{x}$, will converge asymptotically to the true state cannot be satisfied for this choice of A_0. From [16], the maximal value of

$$\frac{a}{\|P\|}$$

as defined in (41) occurs when the matrix Q is the identity matrix which yields

$$\|P\| = 1.809.$$

The Lipschitz constant for the function, $f(x)=\sin x$, is given by the relationship

$$L = 1.$$

Hence, the sufficient condition (41) cannot be met for this choice of A_0. Yet, as demonstrated by the simulations, the observer estimate converges to the true state which indicates that condition (41) is far from necessary.

7.4 The VSS Technique

For the VSS observer we will again consider the system dynamics as represented in equation (49). Again, we will choose the spectrum $\{-1, -1\}$ for the linear part of the observer. Thus,

$$A-KC \triangleq A_0 = \begin{bmatrix} -1 & 0 \\ -1 & -1 \end{bmatrix}. \tag{51}$$

Next, we must find a matrix Q and function $h(\cdot)$ to satisfy assumption #2. One may easily verify that

$$Q = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \tag{52}$$

and

$$-h(t,x) = \sin x_1$$

meet the stipulations of #2. The corresponding positive definite matrix P which satisfies the Lyapunov equation, $A_0^T P + PA_0 = -Q$ is

$$P = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

Therefore, in the notation of assumption #3, $\varsigma(x,t) = -\sin x_1$ and $||\sin x_1|| \leq 1$ and the observer obeys

$$\begin{bmatrix} \dot{\hat{x}}_1 \\ \dot{\hat{x}}_2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x} \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \end{bmatrix} y + S(\hat{x},y) \tag{53}$$

where

$$S(\hat{x},y) = \begin{bmatrix} 0 \\ 2\mathrm{sgn}(\hat{x}_1 + \hat{x}_2 - y) \end{bmatrix}.$$

An estimate for the speed of convergence of our observer is obtained from the minimum eigenvalue of $P^{-1}Q$ $= \min [1,3] = 1$. Therefore, the observer output will converge to the true estimate with a rate which is at least as fast as e^{-t}.

7.5 *Results/Conclusions*

The four observers described by were simulated on a Vax Dual 11/780 utilizing a fifth order Runge-Kutta integration scheme for the initial condition $x_1(0) = \frac{\pi}{2}$ and $x_2(0) = \hat{x}_1(0) = \hat{x}_2(0) = 0$. Figure 1 displays the norm of the error, $||x-\hat{x}||$, as a function of time for each design. From this figure we see that although initially the VSS observer produces the largest amount of error after less than a second the VSS observer error is the smallest of the four. The VSS observer error remains the least in magnitude followed closely by the Lie Algebraic alternative strategy. This is consistent with the theory which guarantees that the quadratic, e^TPe, will converge to zero as fast as e^{-t} for the VSS observer.

Another point in favor of the VSS observer design strategy is ease of implementation. It can be shown that the nonlinear function, $S(\hat{x},y)$ may be replaced by a nonlinear vector-valued function, the elements of which are merely the product of a scalar function, $\rho(t)$, and a signum function of simple linear combinations of $\hat{x}$ and y. Thus, the VSS observer may be easily implemented on a microprocessor. The other three observer dynamics will, in general, be far more nonlinear and not suited for microprocessor implementation.

In summary, let us catalog the advantages and disadvantages of each observer design technique.

The Lie Algebraic method is very attractive in that the system is transformed into a canonical form where observer design is simple and the transformed error dynamics are linear. However, finding a nonlinear transformation which places the system into observer form requires the integration of n coupled partial differential equations. Finally, the Lie Algebraic method requires that knowledge of the nonlinear dynamics of the system be precise.

The method of extended linearization produces an observer error equation which, when linearized about constant operating points of the system, possesses locally invariant eigenvalues. This method also requires the integration of n differential equations and knowledge of the system dynamics must be certain. An apparent advantage of the method of extended linearization over the Lie Algebraic method is that systems which are nonlinear in the scalar input, u, can be accommodated by the extended linearization method, but not by the Lie Algebraic technique. However, Krener and Respondek [8] offer a modified Lie Algebraic approach which can employed for systems which are nonlinear in the control. Finally, the method of extended linearization requires that the system dynamics be known exactly.

The Thau observer guarantees that the observer error is globally asymptotically convergent to zero. Yet, as illustrated in the computer example satisfying the sufficient condition given in (41) which gives such a guarantee may not be possible, even when the system nonlinearities are merely sinusoidal. Also, the Thau observer incorporates the nonlinearities of the system into the observer design, thus certain knowledge of these nonlinearities is essential.

The observer which exhibited the best performance in our example is the VSS observer. This observer guarantees global exponential convergence of a the nonnegative scalar function, e^TPe, without incorporating the system's nonlinearities into the design. One argument against the VSS observer is that it cannot accommodate systems which are nonlinear in the control, as can the modified Lie Algebraic and extended linearization methods. However, with the following two revisions of assumptions 2 and 3 in Section 6, the VSS observer may be applied to systems which are nonlinear in m inputs.

Consider the system

$$\dot{x} = Ax + f(t,x,u)$$

with output

$$y = Cx$$

and further consider the following revisions of the last two assumptions presented in Section 6:

2. There exists a positive definite matrix $Q \in \mathbf{R}^{n \times n}$ and function $h(\cdot)$ where $h(\cdot):\mathbf{R}^1_+ \times \mathbf{R}^n \times \mathbf{R}^m \rightarrow \mathbf{R}^p$ such that

$$f(t,x,u) = P^{-1}C^Th(t,x,u)$$

where P is the unique positive definite solution to the Lyapunov equation

$$PA_0 + A_0^TP = -Q$$

3. There exists a nonnegative function ρ where $\rho(\cdot,\cdot):\mathbb{R}^1_+ \times \mathbb{R}^m \to \mathbb{R}^1_+$ such that

$$||h(x,t,u)|| \leq \rho(t,u) \quad \forall t \in \mathbb{R}^1_+, \ \forall x \in \mathbb{R}^n, \text{ and } \forall u \in \mathbb{R}^m$$

Under these two modified assumptions, the observer output, governed by equation (40), will converge asymptotically to the true state vector, x(t). Thus, the VSS observer design technique can be applied to time varying, multi-input, uncertain systems which are nonlinear in both the states and the control.

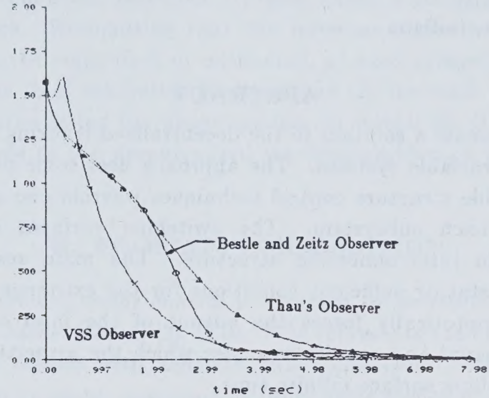


Figure 1: Observer error, $||x - \hat{x}||$.

References

[1] S.H. Żak and C.A. MacCarley, "State-feedback control of nonlinear systems," *Int. J. Control,* Vol. 43, No. 5, pp. 1497-1515, 1986.

[2] R. Marino, "High-gain feedback in nonlinear control systems," *Int. J. Control,* Vol. 42, No. 6, 1369-1385, 1985.

[3] L.R. Hunt, R. Su, and G. Meyer , *Differential Geometric Control Theory,* edited by W. Brockett, Birkhauser, Boston, 1983.

[4] A.J. Van Der Schaft, "On nonlinear observers," *IEEE Trans. on Automat Contr.,* Vol. AC-30, No. 12, pp. 1254-1256.

[5] S.R. Kou, D.L. Elliot, and T.J. Tarn, "Exponential observers for nonlinear dynamic systems," *Inform. Contr.,* Vol. 29, pp. 204-216, 1976.

[6] J.J. Slotine, J.K. Hedrick, and E.A. Misawa, "On sliding observers for nonlinear systems," *Proc. of ACC,* pp. 1794-1800, June 1986.

[7] D. Bestle and M. Zeitz, "Canonical form observer design for nonlinear time-variable systems," *Int. J. Control,* Vol. 38, No. 2, pp. 419-431, 1983.

[8] A. Krener and W. Respondek, "Nonlinear observers with linearizable error dynamics," *SIAM J. Contr. and Optimization,* Vol. 23, No. 2, pp. 197-216, 1985.

[9] F.E. Thau, "Observing the state of nonlinear dynamic systems," *Int. J. Control,* Vol. 17, pp. 471-479, 1973.

[10] W. Baumann and W. Rugh, "Feedback control of nonlinear systems," *IEEE Trans. Automat. Contr.,* Vol. AC-31, No. 1, pp 40-47, 1986.

[11] B.L. Walcott and S.H. Żak, "Observation of dynamical systems in the presence of bounded nonlinearities/uncertainties," *Proc. of CDC,* 1986.

[12] B.L. Walcott, M.J. Corless and S.H. Żak, "Comparative study of nonlinear state observation techniques," *Int. J. Control,* (to appear).

[13] J.E. Ackermann, "On the synthesis of linear control systems with specified characteristics," *Automatica,* Vol. 13, No. 1, pp. 89-93, 1977.

[14] J.C. Willems, *Stability Theory of Dynamical Systems,* John Wiley & Sons, New York, 1970.

[15] A. Steinberg and M. Corless, "Output feedback stabilization of uncertain dynamical systems," *IEEE Trans. Automat. Contr.,* Vol. AC-30, No. 10, pp. 1025-1027, 1985.

[16] R.V. Patel and M. Toda, "Quantitative measures of robustness for multivariable systems," *Proc. of JACC,* p. TP8-A, 1980.

A VARIABLE STRUCTURE CONTROL APPROACH TO THE NONLINEAR DECENTRALIZED PROBLEMS

GREGORY P. MATTHEWS[†] AND RAYMOND A. DECARLO
School of Electrical Engineering
Purdue University
West Lafayette, Indiana

ABSTRACT

This paper presents a solution to the decentralized tracking problem for a class of nonlinear, multivariable systems. The approach uses some previously developed decentralized variable structure control techniques wherein one associates a switching surface with each subsystem. The switching surfaces are interconnected through the system interconnection structure. The main results of the paper include a theorem stating sufficient conditions for the existence of a decentralized control which asymptotically forces the output of the interconnected system to track a reference signal and conditions under which the associated state trajectory intercepts the switching surface infinite time.

I. INTRODUCTION

The purpose of this paper is to present a solution to the decentralized servomechanism problem for a class of interconnected, nonlinear systems having multiple input, multiple output (MIMO) subsystems. Specifically the paper extends the decentralized variable structure control (DVSC) theory developed in [1] to the tracking problem.

The first step in our design procedure is the selection of a set of special local manifolds (sliding surfaces) which are linear in the subsystem states, local interconnection inputs, and reference states. These local sliding surfaces are constructed such that the equivalent system, which results from the restriction of the original subsystems to the subsystem sliding surfaces, has the desired tracking behavior.

As shown in section 3, under appropriate conditions the linear interconnected sliding surfaces may be designed via linear decentralized control techniques: by viewing the equations of the sliding surfaces as a linear decentralized feedback control law. Since DVSC provides a mechanism of forcing a class of interconnected nonlinear systems to act as linear, time invariant interconnected systems [1], one possible method of determining the sliding surface parameters (sliding surface construction) is to use Davison's [7,15,16,17] method of decentralized tracking for linear, time invariant systems. Rather than extending the DVSC sliding surface design developed in [1] to work with Davison's method, a new state feedback approach to the linear, time invariant, decentralized tracking or servomechanism problem was developed. The technique is an extension of Kwatney and Kalnitsky's *Deviation and Steady State Controls* [8] for global systems. This method was also used by Young and Kwatney [5] for global VSC of linear, time invariant systems, and is based on the feedback of the system and reference system's state variables. To make the extension to decentralized control, decentralized eigenvalue assignment [9,10] using subsystem state feedback replaces global eigenvalue assignment using plant state feedback. Recognizing that the reference signal is known apriori, the reference states can be computed, or edtimated, at each subsystem. Thus, the control depends only on local subsystem feedback and the feedback of locally generated reference states. After using the above method to extend the DVSC sliding surface construction methods to the decentralized tracking problem, a simple DVSC law is presented.

[†] Currently with the Electronics Department of General Motors Research Laboratories, Warren, MI 48090.

II. PROBLEM FORMULATION

As in [1] the system under control is assumed to be represented by the *Component Connection Model* (CCM) [2]. The CCM represents the systems as a series of N subsystem state models with algebraic interconnections. The i-th subsystem is assumed to have an m_i-input, p_i-output, n_i order state model of the form

$$\begin{aligned} \dot{x}_i(t) &= A_i(x_i,t)x_i(t) + B_i[a_i(t) + u_i(t)] \\ b_i(t) &= C_i x_i(t) + D_i a_i(t) \,. \end{aligned} \tag{2.1}$$

Linear algebraic subsystem and output interconnections take the form

$$\begin{aligned} a(t) &= L_{11}b(t) \\ y(t) &= L_{21}b(t) \end{aligned} \tag{2.2}$$

where $x_i \epsilon R^{n_i}$, $a_i \epsilon R^{m_i}$, $u_i \epsilon R^{m_i}$, $b_i \epsilon R^{p_i}$ are the ith subsystem's state, interconnection input, feedback (control) input, and output vectors respectively. Further, $a = col[a_1,...,a_N]$ and $b = col[b_1, . . . , b_N]$ are the composite interconnection input and output vectors while $y \epsilon R^r$ is the system output. In addition the following assumptions are made:

(A1) Every state $x_i^j(t)$ can be locally observed;

(A2) The pair $(A_i(x_i,t)x_i,B_i)$ is controllable in the large;

(A3) The matrix B_i is full rank and has only constant entries;

(A4) Each subsystem is input linearizable [1];

(A5) For each fixed feedback structure each subsystem is Lipschitz;

(A6) The interconnection inputs a_i^j can be locally observed.

The objective is to find a decentralized control such that the system output $y(t) \rightarrow y^*(t)$, as $t \rightarrow \infty$, where $y^*(t)$ is the output of the q dimensional linear, time invariant reference system

$$\begin{aligned} \dot{w}(t) &= Zw(t) \\ y(t)^* &= Gw(t) \,. \end{aligned} \tag{2.3}$$

To use DVSC theory as presented in [1], the $n-m$ dimensional sliding surface is defined by

$$\sigma = \mathrm{col}[\sigma_1, \ldots, \sigma_N] = 0$$

where $x = \mathrm{col}[x_1,\ldots,x_N]$,

$$\sigma_i(x_i,a_i,w) = S_i x_i - S_i B_i \int_{-\infty}^{t} a_i(q)dq + K_g^i w \tag{2.4}$$

with $S_i \epsilon R^{m_i \times n_i}$, $\sigma_i \epsilon R^{m_i}$, $K_g^i \epsilon R^{m_i \times q}$, while the remaining terms are as previously defined. The first term $S_i x_i$ in equation (2.4) is the conventional linear manifold term [3], while the integral term arises from the decentralized control structure imposed on the interconnected system [1], and the term $K_g^i w$ is added for asymptotic tracking.

III. SLIDING SURFACE DESIGN

The first phase of DVSC is the construction of a set of interconnected local sliding surfaces to produce an equivalent linear, time invariant system whose interconnection structure is consistent with that of the plant and whose behavior meets the objective of asymptotic tracking, [1]. This is accomplished after first converting each subsystem to the so called regular form [19]. For convenience, the method of equivalent control [3,4] is avoided. Assumptions (A3) and (A4) insure that the required transformations are linear and time invariant [1].

Based on the structure of B_i let a series of local state transformations ($z_i = T_i x_i$, $i = 1,\ldots,N$) be such that the m_i local inputs only affect the last m_i states. Specifically, choose $T_i \epsilon R^{n_i \times n_i}$ such that

$$T_i B_i = \begin{bmatrix} 0 \\ B_i^2 \end{bmatrix} \tag{3.2}$$

where B_i^2 is an $m_i \times m_i$ matrix of full rank by (A3), and

$$z_i = \begin{bmatrix} z_i^1 \\ z_i^2 \end{bmatrix}, \quad T_i A_i T_i^{-1} = \begin{bmatrix} A_i^{11} & A_i^{12} \\ A_i^{21} & A_i^{22} \end{bmatrix}$$

From equations (3.2) and (2.1), the subsystem dynamics in the new coordinates are

$$\dot{z}_i^1 = A_i^{11} z_i^1 + A_i^{12} z_i^2 \tag{3.3a}$$

$$\dot{z}_i^2 = A_i^{21}(z_i,t) z_i^1 + A_i^{22}(z_i,t) z_i^2 + B_i^2 [a_i + u_i] \tag{3.3b}$$

$$b_i = C_i T_i^{-1} z_i + D_i a_i = C_i^1 z_i^1 + C_i^2 z_i^2 + D_i a_i \tag{3.4}$$

where $z_i^1 \epsilon R^{n_i - m_i}$, $z_i^2 \epsilon R^{m_i}$ and from assumption (A4) the matrices A_i^{11} and A_i^{12} are constant [1]. In the new or z-coordinates the local sliding surface (2.4) is given by

$$\sigma_i(z_i^1, z_i^2, a_i, w) = S_i^1 z_i^1 + S_i^2 z_i^2 + K_g^i w - S_i^2 B_i^2 \int_{-\infty}^{t} a_i(q)dq = 0, \tag{3.5}$$

where $S_i T_i^{-1} = [S_i^1 \,|\, S_i^2]$. Solving for z_i^2 and inserting in (3.3a) yields the local equivalent subsystems

$$\begin{aligned} \dot{z}_i^1 &= \left[A_i^{11} - A_i^{12} (S_i^2)^{-1} S_i^1 \right] z_i^1 - A_i^{12} (S_i^2)^{-1} K_g^i w + A_i^{12} B_i^2 \tilde{a}_i \\ \bar{b}_i &= C_i T_i^{-1} \int_{-\infty}^{t} z_i(q)dq + D_i \tilde{a}_i \end{aligned} \tag{3.6}$$

where $\tilde{a}_i = \int a_i(q)dq$, $\bar{b}_i = \int b_i(q)dq$, and the interconnection structure is given by $\tilde{a} = L_{11} \bar{b}$.

To insure the well posedness of (3.6) it is required that $\bar{b}_i$ depend only on z_i^1, $i = 1,\ldots,N$, [18]. This often requires the addition of local dynamic compensation. To simplify the exposition assume that appropriate local dynamic compensation has been added according to the procedure described in [1,B]. Then there exist $m_i \times n_i$ matrices G_i, $i = 1,\ldots,N$, such that

$$\int_{-\infty}^{t} z_i(q)dq = G_i z_i^1 \tag{3.7}$$

producing the well posed dynamical subsystem models

$$\begin{aligned} \dot{z}_i^1 &= \left[A_i^{11} - A_i^{12} (S_i^2)^{-1} S_i^1 \right] z_i^1 - A_i^{12} (S_i^2)^{-1} K_g^i w + A_i^{12} B_i^2 \tilde{a}_i(q) \\ \bar{b}_i &= C_i T_i^{-1} G_i z_i^1 + D_i \tilde{a}_i \end{aligned} \tag{3.8}$$

where

$$\tilde{a}_i = \int_{-\infty}^{t} a_i(q)dq, \quad \bar{b}_i = \int_{-\infty}^{t} b_i(qA)dq \,.$$

The interconnection structure is given by

$$\tilde{a} = L_{11} \bar{b}$$

$$\tilde{y} = L_{21} \bar{b} \,.$$

The tracking problem, however, requires that $y = \dot{\tilde{y}} \rightarrow y^*$. Since $y = L_{21} b$, the solution of this tracking problem (as will become clear) necessitates expressing each b_i as a linear function of z_i^1, $\tilde{a}_i$, and w. Some straightforward algebra reveals that on the sliding surface

$$b_i = \left[C_i^1 - C_i^2(S_i^2)^{-1}\right]z_i^1 + C_i^2B_i^2\tilde{a}_i - C_i^2(S_i^2)^{-1}K_g^i w \tag{3.10}$$

The forthcoming solution of the tracking problem further requires knowledge of the state matrices of the composite system model. To construct this composite representation let

$$A^{11} = \text{block} - \text{diag}\,[A_1^{11},\ldots,A_N^{11}],\ \ A^{12} = \text{block} - \text{diag}\,[A_1^{12},\ldots,A_N^{12}]\,,$$

$$B^2 = \text{block} - \text{diag}\,[B_i^2,\ldots,B_N^2],\ \ S^i = \text{block} - \text{diag}\,[S_1^i,\ldots,S_N^i]\,;\quad (i=1,2),$$

$$C^1 = \text{block} - \text{diag}\,[C_1^1,\ldots,C_N^1],\ \ C^2 = \text{block} - \text{diag}\,[C_1^2,\ldots,C_N^2]\,,$$

$$D = \text{block} - \text{diag}\,[D_1,\ldots,D_N],\ \ CT^{-1} = [C^1 \,|\, C^2]$$

$$G = \text{block} - \text{diag}\,[G_1,\ldots,G_N],\ \ T = \text{block} - \text{diag}\,[T_1,\ldots,T_N]$$

and

$$K_g = \begin{bmatrix} K_g^1 \\ \vdots \\ K_g^N \end{bmatrix}.$$

Using equations (3.7), (3.8), (3.10), and the identification $v = S^1z^1 + K_gw$, one obtains a composite representation of the equivalent interconnected system as

$$\begin{aligned} \dot{z}^1 &= F_{eq}z^1 + B_{eq}v \\ y &= C_{eq}z^1 + E_{eq}w \end{aligned} \tag{3.11}$$

where if $\bar{L}_{11} = (I - L_{11}D)^{-1}L_{11}CT^{-1}G$:

$$F_{eq} = A^{11} + A^{12}B^2\bar{L}_{11} \tag{3.12a}$$

$$B_{eq} = -A^{12}(S^2)^{-1} \tag{3.12b}$$

$$C_{eq} = L_{21}(I - DL_{11})^{-1}\left[C^1 - C^2(S^2)^{-1}S^1 + C^2B^2\bar{L}_{11}\right] \tag{3.12c}$$

$$E_{eq} = -L_{21}(I - DL_{11})^{-1}C^2(S^2)^{-1}K_g. \tag{3.12d}$$

The composite system input $v = S^1z^1 + K_gw$ is viewed as a linear feedback control which is decentralized in the subsystem states.

Sufficient conditions for tracking to occur on the sliding surface are given by the following theorem.

Theorem 3.13: Given the interconnected composite system represented by (3.11) as derived from (3.8) and (3.9), assuming (A1)-(A6) are satisfied, a set of conditions sufficient for the existence of a "control" $v = S^1z^1 + K_gw$, such that $y \longrightarrow y^*$, where $S^1 = \text{block} - \text{diag}\,[S_1^1,\ldots,S_N^1]$, are: (1) at most one subsystem is of odd dimension; (2) there exists an $n \times q$ matrix X and an $m \times q$ matrix U satisfying

$$\begin{aligned} &\text{(i)}\ \ XZ - F_{eq}X = B_{eq}U \\ &\text{(ii)}\ \ C_{eq}X = G - E_{eq} \end{aligned} \tag{3.14}$$

Further, $K_g = U - S^1X$.

Proof: Define two signals $\bar{z}^1 = Xw$ and $\bar{v} = Uw$ where X and U satisfy (3.14). Now define the differential variables $\Delta z^1 = z^1 - \bar{z}^1$, and $\Delta v = v - \bar{v}$. Given these definitions a little algebra shows that the error

$$e = y - y^* = C_{eq}\Delta z^1 \tag{3.15}$$

If Δz^1 is asymptotically stable tracking will occur if v has the proper form. The remainder of the proof has two parts. First, we show that Δz^1 can be made exponentially stable and secondly that the control v has the proper form.

Again, given the above definitions, some straightforward algebra shows that

$$\Delta\dot{z}^1 = F_{eq}\Delta z^1 + B_{eq}\Delta v \tag{3.16}$$

where $\Delta v = S^1\Delta z^1$. From [9,10] and [1], if assumptions (A1)-(A6) and condition (1) are satisfied, then there exists a feedback matrix

$$S^1 = \text{block} - \text{diag}\,[S_1^1, \ldots, S_N^1] \tag{3.17}$$

that assigns any desired (complex symmetric) spectrum to (3.16) implying the existence of a stabilizing matrix. Hence $\Delta z^1 \to 0$ so that $z^1 \to \bar{z}^1$, which implies that $y \to y^*$ as $t \to \infty$.

From the above development the control v which guarantees tracking must have the form

$$v = \Delta v + \bar{v} = S^1\Delta z^1 + Uw \tag{3.18}$$

where S^1 has the block structure of (3.17) and stabilizes (3.16). It remains to properly define K_g so that the given control $v = S^1z^1 + K_gw$ is consistent with (3.18). In this task, observe that $\Delta z^1 = z^1 - \bar{z}^1 = z^1 - Xw$, so that

$$v = S^1\Delta z^1 + Uw = S^1z^1 + (U - S^1X)w = S^1z^1 + K_gw \tag{3.19}$$

Thus, if $K_g = U - S^1X$, then v has the proper form.

Note, that if necessary, the condition (1) can be satisfied by addition of local dynamic compensation. Further, condition (1) is only required for arbitrary eigenvalue assignment. Provided that the system represented by equation (3.16) has no unstable decentralized fixed modes, then decentralized stabilization techniques, such as those of Wang and Davison [21] or Sezer and Huseyin [22], may be used to find a stabilizing control $\Delta v = S^1\Delta z^1$.

IV. CONTROL DESIGN

The next step is to select a VS control strategy which drives the system's state trajectory to and maintains it on the sliding surface. In the ideal switching case (considered here) the asymptotic system behavior is independent of the control used, provided that a sliding mode is established. In the following, a simple control strategy is presented which is valid for the ideal case and has been demonstrated via simulation [11] in the case where the switching speed is significantly faster than the plant's response time. A simple and effective control law is

$$u_i = -(S_iB_i)^{-1}\left[S_iA_i(x_i)x_i + K_g^iZw + \Gamma_iSGN(\sigma_i)\right] \tag{4.1}$$

where $\Gamma_i = \mathrm{diag}[\gamma_i^1,...,\gamma_i^{m_i}]$ with $\gamma_i^j > 0$, $j = 1,...,m_i$, and $SGN(\sigma_i) = \mathrm{col}[\mathrm{sgn}(\sigma_1),...,\mathrm{syn}(\sigma_{m_i})]$, $i = 1,...,N$.

At this point it should be noted that the control (4.1) is decentralized in the subsystem states. Further, since w(t) is a known set of signals which can be reproduced (or estimated from $y^*(t)$) at each subsystem, the control can be determined locally at each subsystem.

To verify that this control law drives the system to and maintains it on the sliding surface we show that the system with this control satisfies a sufficient condition for the existence of a sliding mode, [3].

Theorem 4.2: The interconnected system (2.1), (2.2) with sliding surface (2.4) and control law (4.1) satisfies the sufficient condition for the existence of a sliding mode:

$$\sigma_i^T\dot{\sigma}_i < 0, \tag{4.3}$$

for $i = 1,..,N$, when $\sigma_i^T\sigma_i \neq 0$.

Proof:

$$\dot{\sigma}_i = S_i\dot{x}_i + K_g\dot{w} - S_iB_ia_i = S_iA_i(x_i)x_i + S_iB_iu_i + K_g^iZw = -\Gamma_iSGN(\sigma_i) \tag{4.4}$$

Hence

$$\sigma_i^T\dot{\sigma}_i = -\sigma_i^T\Gamma_iSGN(\sigma_i) = -\sum_{j=1}^{m_i}\gamma_i^j\,|\sigma_i^j|\,.$$

If all γ_i^j are positive, as assumed, then the above is negative off of the surface. The next important consideration, is the time at which the trajectory actually intercepts the intersection of all the switching surfaces, forcing a sliding mode behavior.

Theorem 4.5: The time it takes to intercept the j-th switching surface of the i-th subsystem is finite and inversely proportional to γ_i^j.

Proof: $\dot{\sigma}_i^j = -\gamma_i^j\,\mathrm{sgn}(\sigma_i^j)$. Thus if the starting time is t_0 and if $\sigma_i^j(x_i(t_0)) > 0$, then $\sigma_i^j(t) = \sigma_i^j(x_i(t_0)) - \gamma_i^j(t - t_0)$. So the time ($t_i^j$) to intercept $\sigma_i^j = 0$ is

$$t_i^j = \frac{|\sigma_i^j(x_i(t_0))|}{\gamma_i^j}\,. \tag{4.6}$$

Equation (4.6) also holds if $\sigma_i^j(x_i(t_0)) < 0$.

Corollary 4.7: The time to intercept the intersection of all m_i switching surfaces of the i-th subsystem is given by

$$t_i = \max \left(\frac{|\sigma_i^1(x_i(t_0))|}{\gamma_i^1}, \ldots, \frac{|\sigma_i^{m_i}(x_i(t_0))|}{\gamma_i^{m_i}} \right). \tag{4.8}$$

Corollary 4.9: The time to intercept the intersection of all switching surfaces is given by

$$t^+ = \max(t_1, t_2, \ldots, t_N) \tag{4.10}$$

where t_i, $i = 1,\ldots,N$ are given by (4.8). Thus, assuming that the initial states are finite and that each $\gamma_i^j > 0$, a sliding mode is reached in finite time t^+ which can be reduced by increasing γ_i^j, $j = 1,\ldots,m_i$, $i = 1,\ldots,N$.

Even if there are parameter variations in $A_i(x_i)$ or input disturbances, provided they are bounded and in the image of B_i, by selecting γ_i^j sufficiently large a sliding mode still exists and is still reachable [3,6,12]. However, it may not be possible to reach it in finite time. In practice this rarely causes any difficulty since the trajectory is within a small distance of the sliding surface after some finite time. However, large values of γ_i^j lead to large control discontinuities and thus to chattering. To avoid excessive chattering while still providing the desirable robustness and short reaching times an adaptive selection of γ_i^j similar to [13] or [11] is suggested.

VI. REFERENCES

1. G.P. Matthews and R.A. DeCarlo, "Decentralized variable structure control for class of nonlinear multi-input multi-output nonlinear systems," *Proced. of the -th Annual IEEE Conf. on Decision and Control,* 1985.
2. R.A. DeCarlo and R. Saeks, *Interconnected Dynamical Systems,* Marcel Dekker, New York, 1981.
3. V.I. Utkin, *Sliding Modes and Their Application in Variable Structure systems,* MIR Publishers, Moscow, 1978.
4. V.I. Utkin, "Equations of the slipping regime in discontinuous systems I," *Automat. Remote Contr.,* vol. 33, pp. 211-219, 1972.
5. K-K.D. Young and H.G. Kwatny, "Variable structure servomechanism design and applications to overspeed protection control," *Automatica,* pp. 385-400, July 1982.
6. B. Drazenonic, "The invariance condition in variable structure systems," *Automatica,* vol. 5, 1969.
7. E.J. Davison and A. Goldenberg, "Robust control of a general servomechanism problem: the servo compensator," *Automatica,* vol. 11, pp. 461-471, 1975.
8. H.G. Kwatny and K.C. Kalnitsky, "On alternative methodologies for the design of robust linear multivariable regulators," *IEEE Trans. Automat. Contr.,* vol. AC-23, No. 5, 1978.
9. S. Richter, R. DeCarlo, "A Homotopy method for eigenvalue assignment using decentralized state feedback," IEEE *Trans. Automat. Contr.,* vol. AC-28, No. 5, pp. 148-159, 1984.
10. S. Richter, S. Lefebvre, R. DeCarlo, "A continuation algorithm for eigenvalue for eigenvalue assignment by decentralized constant-output feedback," *Int. Journal of Control,* vol. 41, Vol. 5, pp. 1273-1292, 1985.

11. G.P. Matthews, "Variable Structure Control of Interconnected Nonlinear Systems," PhD. Thesis, Purdue University, 1985.

12. E. P. Ryan and M. Corless, "Ultimate boundedness and asymptotic stability of a class of uncertain dynamical systems via continuous and discontinuous feedback control," *IMA Jrnl of Math. Control and Information,* vol. 1, pp. 223-242, 1984.

13. R.G. Morgan and U. Ozgunner, "A decentralized variable structure control algorithm for robotic manipulators," *IEEE Robotics and Automat.,* vol. RA-1, pp. 57-65, March 1985.

14. G.H. Goulb and C.F. Van Loan, *Matrix Computations,* John Hopkins University Press, Baltimore, Maryland, 1983.

15. E.J. Davison, "Decentralized stabilization and regulation in large multivariable systems," in Y.C. Ho and S.K. Mitter, Eds. *Direction in Large-Scale Systems,* pp. 303-323, Plenum Press, New York.

16. E.J. Davison, "The robust decentralized control of a general servomechanism problem," *IEEE Trans. Automat. Cont.,* AC-21, pp. 14-24, 1976.

17. E.J. Davison, "The robust decentralized servomechanism problem for composite systems with input-output interconnections," *IEEE Trans. Automat. Cont.,* AC-24, pp. 325-327, 1979.

18. G.P. Matthews and R.A. DeCarlo, "Decentralized variable structure control for class of nonlinear multi-input multi-output nonlinear systems," Submitted to *Circuits, Systems, and Signal Processing.*

19. A.G. Luk'yanov and V.I. Utkin, "Methods of reducing equations of dynamic systems to regular form," *Avtomat. Telemekh.,* No. 4, pp. 5-13.

20. W.M. Wonham, *Linear Multivariable Control: A Geometric Approach,* Second Edition, Springer-Verlag, New York, 1979.

21. Wang and Davison, "On the stabilization of decentralized control systems," *IEEE Trans. Auto. Control,* Vol. AC-18, pp. 473-478, Oct. 1970.

22. Sezer and Huseyin, "Stabilization of linear interconnected systems using local state feedback," *IEEE Trans. Systems Man and Cybern,* Vol. 8, pp. 751.

DETERMINING STABILITY OF A SLOWLY VARYING NONLINEAR SYSTEM THROUGH A FAMILY OF LINEARIZATIONS

C. L. DEMARCO and B. R. BARMISH
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
Madison, Wisconsin 53706

ABSTRACT

This paper considers the relationship between the behavior of a nonlinear system $\dot{x}(t)=f(x(t),u(t))$ and the stability of a family of linear systems obtained via linearization about operating points corresponding to "frozen" values of the input. Subject to certain regularity assumptions on $f(\cdot,\cdot)$, it is shown that the state of the nonlinear system can be kept in an arbitrarily small neighborhood of the time varying operating point provided the family of linearizations is strictly stable and $|\dot{u}(t)|$ is sufficiently small. The trade-off between desired tracking accuracy and time rate of change of the input is explicitly displayed.

BACKGROUND AND MOTIVATION

In recent years, considerable attention has been devoted to the problem of simultaneous stabilization of a family of linear plants; e.g., see [1] and its bibliography. One source of motivation for such work is derived from the intuitively appealing concept that it may be possible to guarantee stability of a time varying nonlinear system by stabilizing an "associated" family of linear systems. However, while the idea of performing controller design for nonlinear systems based on the linearization at a sequence of operating points has often been successful in practice (consider the wide spread use of gain scheduling), the relationship between the stability of the family of linearizations, the nature of the trajectory of operating points, and the underlying nonlinear system has received relatively little attention in the literature. Existing results addressing some aspects of this problem may be found in [2] and [3], based on work originally reported in [4].

A more general version of the problem above involves tracking; i.e., one wishes to guarantee that the state of a system "tracks" a desired time varying operating point. While a number of works exist that treat the nonlinear tracking problem (see, for example, [5], [6], and [7]), the results in these papers are primarily asymptotic in nature and give conditions under which the output of a system will track the input as $t \to \infty$. Such asymptotic results are clearly not suitable for determining if the state of the system remains in a neighborhood of the desired operating point over the entire time horizon of interest.

The goal of this paper is to use the information on the stability of the family of linearizations along with conditions on the trajectory of the operating point to obtain a bound on the maximal error between the state of the nonlinear system and the desired operating point. Under fairly weak assumptions, we show that if the family of linearizations is strictly stable, then for any specified bound $\varepsilon > 0$ for the tracking error $\mathbf{e}(t)$, there exists a bound on the initial error and a bound on the time rate of change of the input guaranteeing that $|\mathbf{e}(t)| < \varepsilon$ for all t.

PROBLEM FORMULATION

We assume that the variation in the operating point of the system is induced by some exogenous input. For example, consider the case of an electric power system. The customers' power demand at each bus of the network is a slowly varying input to the system. The variation in this input induces a drift in the bus voltage magnitudes and phases which in turn determine the operating point of the system. The power system is judged to be "small disturbance stable" if the linearization of the nonlinear model at each operating point is strictly stable. Similar analyses are carried out for a wide class of engineering systems. This characterization of the variation in operating point through its dependence on an external input to the system captures the physical nature of many systems of interest. In addition, it allows a fairly general stucture for the system model. We are concerned with the behavior of systems modeled by state equations of the form:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{1}$$

where $\mathbf{f}: \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $\mathbf{u}: \mathbb{R}_+ \to \Omega \subset \mathbb{R}^m$, Ω being a prespecified input restraint set.

Furthermore, we assume that the system (1) satisfies the following assumptions:

(A1) Smooth $\mathbf{u}(\cdot)$ with compact range:

a) $\mathbf{u}(\cdot)$ is C^1, once continuously differentiable.

b) Ω is a compact subset of $\mathbb{R}^m$.

(A2) Smooth $\mathbf{f}(\cdot,\cdot)$ with bounded partial derivatives:

a) $\mathbf{f}(\cdot,\cdot)$ is C^2.

b) Letting D_1 and D_2 denote differentiation with respect to the first and second arguments of $\mathbf{f}(\cdot, \cdot)$, it is assumed that there exist constants M_{f1} and M_{f2} such that

$$|D_1\mathbf{f}(\mathbf{x}, \omega)| < M_{f1};$$

$$|D_2\mathbf{f}(\mathbf{x}, \omega)| < M_{f2}$$

for all pairs $(\mathbf{x}, \omega) \in \mathbb{R}^n \times \Omega$.

(A3) Smooth dependence of equilibrium on $\omega \in \Omega$: There exists a function $\varphi: \mathbb{R}^m \rightarrow \mathbb{R}^n$ having the following properties:

a) $\varphi(\cdot)$ is C^1.

b) $\mathbf{f}(\varphi(\omega), \omega) = 0$ for all $\omega \in \Omega$.

(A4) Strictly stable linearizations about each operating point: For each $\omega \in \Omega$, all eigenvalues of $D_1\mathbf{f}(\varphi(\omega), \omega)$ have strictly negative real part.

Remarks: Assumptions (A1) and (A2) imply that for each fixed $\omega \in \Omega$, $\mathbf{f}(\cdot, \omega)$ satisfies a global Lipschitz condition and that for each fixed $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{f}(\mathbf{x}, \mathbf{u}(\cdot))$ is continuous. Standard results in ordinary differential equations then ensure that (1) has a unique continuous solution that may be continued indefinitely for positive t [8]. The reader should also note that (A3) allows for the possibility of multiple equilibria for each "frozen" system.

Based on these assumptions, it will prove convenient to develop notation describing the local behavior of the system about the operating point of interest. To this end, the proof of the main result exploits an "error sytem" that describes the deviation in the state from the operating point $\varphi(\mathbf{u}(t))$. Given an input $\mathbf{u}(\cdot)$ and initial condition $\mathbf{x}(t_0)$, and letting

$$\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_0(t)$$

where

$$\mathbf{x}_0(t) = \varphi(\mathbf{u}(t)),$$

it follows that

$$\dot{e}(t) = f(x_0(t)+e(t), u(t)) - \dot{x}_0(t). \tag{2}$$

The reader should note that $x_0(t)$ does not define a trajectory for (1); rather $x_0(t)$ is interpreted as the equilibrium for a frozen system that would result if the input were held constant at the value $u(t)$. The following theorem bounds the magnitude of the tracking error $e(t)$ for the case when the time rate of change of the input $u(t)$ is sufficiently small.

Theorem: Let Assumptions (A1) through (A4) hold. Then for any $\varepsilon > 0$, there exist constants $\delta > 0$ and $M_u > 0$ having the following property: Given any initial time t_0 and any admissible input $u(\cdot)$, if

$$|x(t_0) - \varphi(u(t_0))| < \delta$$

and

$$\sup_{t \geq t_0} |D_t u(t)| < M_u,$$

it follows that the tracking error $e(t)$ is bounded in norm by ε for all time; i.e.,

$$\sup_{t \geq t_0} |x(t, x(t_0), t_0) - \varphi(u(t))| < \varepsilon$$

where $x(t, x(t_0), t_0)$ denotes the trajectory of (1) originating from initial condition $x(t_0)$ at time t_0.

Proof

From (A4), it follows that for each $\omega \in \Omega$, there exists a unique symmetric positive definite matrix $P(\omega)$ satisfying

$$A^T(\omega)P(\omega) + P(\omega)\, A(\omega) = -2I \tag{3a}$$

where

$$A(\omega) \equiv D_1 f(\varphi(\omega), \omega). \tag{3b}$$

Given that $\mathbf{A}(\omega)$ remains nonsingular, the components of $\mathbf{P}(\omega)$ depend continuously on the components of $\mathbf{A}(\omega)$, and from (A2) and (A3), it follows that $\mathbf{P}(\cdot)$ is a C^1 function. Furthermore, letting $\lambda_{\min}(\mathbf{M})$ and $\lambda_{\max}(\mathbf{M})$ denote the minimum and maximum eigenvalues of the real symmetric positve definite matrix $\mathbf{M}$, it follows from the compactness of Ω and the continuity of $\mathbf{P}(\omega)$ that there exist real constants $\mu_{\min} > 0$ and $\mu_{\max} < \infty$ such that

$$\lambda_{\min}(\mathbf{P}(\omega)) > 2\mu_{\min} \text{ for all } \omega \in \Omega;$$

$$\lambda_{\max}(\mathbf{P}(\omega)) < 2\mu_{\max} \text{ for all } \omega \in \Omega.$$

Therefore, we may conclude

$$\mu_{\min} |\mathbf{e}|^2 < 1/2\{\mathbf{e}^T\mathbf{P}(\omega)\mathbf{e}\} < \mu_{\max} |\mathbf{e}|^2 \text{ for all } \omega \in \Omega.$$

Our goal is to employ $V(\mathbf{e},t)=1/2\{\mathbf{e}^T\mathbf{P}(\mathbf{u}(t))\mathbf{e}\}$ as a candidate Lyapunov function for (2) to ensure the boundedness (though not stability) of $\mathbf{e}(t)$ under suitable conditions on M_u. Examining the derivative of $V(\mathbf{e},t)$ along trajectories of (2) yields

$$\dot{V}(\mathbf{e},t) = \mathbf{e}^T\mathbf{P}(\mathbf{u}(t))\left[\mathbf{f}(\mathbf{x}_0(t) + \mathbf{e}, \mathbf{u}(t))\right] - \mathbf{e}^T\mathbf{P}(\mathbf{u}(t)\left[D_u\varphi(\mathbf{u}(t)) \cdot D_t\mathbf{u}(t)\right]$$
$$+ 1/2\sum_{i=1}^{n}\sum_{j=1}^{n} D_u p_{ij}(\mathbf{u}(t))D_t\mathbf{u}(t)\, e_i e_j \,. \tag{4}$$

Noting that $D_u p_{ij}(\cdot)$ is continuous, the compactness of Ω is again invoked to ensure that there exists a constant $M_V < \infty$ such that

$$1/2\sum_{i=1}^{n}\sum_{j=1}^{n} D_u p_{ij}(\mathbf{u}(t))D_t\mathbf{u}(t)\, e_i e_j < M_V \cdot M_u |\mathbf{e}|^2$$

Likewise, $D_u\varphi(\cdot)$ is continuous, so there exists a constant $M_\varphi < \infty$ such that

$$|D_u\varphi(\omega)| \leq M_\varphi \quad \text{for all } \omega \in \Omega.$$

Furthermore, we observe that

$$f(x_0(t)+e(t),\ u(t)) = A(u(t))e(t) + \theta(e(t))$$

where $\theta(e)$ denotes the remainder term in the Taylor expansion which is $o(|e|)$ ("order $|e|$"); i.e.,

$$\lim_{|e|\to 0} \frac{\theta(e)}{|e|} = 0.$$

Substituting these results into (4) yields

$$\dot{V}(e, t) \le -|e|^2 + e^T P(u(t))\cdot\theta(e) + 2M_u M_\varphi \mu_{max} |e| + M_v M_u |e|^2$$

$$= -(1 - M_v M_u)|e|^2 + o(|e|^2) + 2M_u M_\varphi \mu_{max} |e|. \tag{5}$$

By definition of the $o(\cdot)$ function, for any $\gamma > 0$ there exists an $r > 0$, $r \le \varepsilon$, such that $o(|e|^2) \le \gamma |e|^2$ for $|e| < r$. In particular, for $\gamma = 1/2$, we conclude that there exists an r, $0 < r \le \varepsilon$, such that

$$V(e, t) \le -(1/2 - M_v M_u)|e|^2 + 2M_u M_\varphi \mu_{max} |e| \tag{6}$$

for all $|e| < r \le \varepsilon$.

Next, let $\delta < (\mu_{min}/\mu_{max})^{1/2} r$, and choose M_u such that

$$M_u < \frac{\delta}{2(M_v\delta + 2M_\varphi \mu_{max})} \tag{7}$$

We now show that if $|e(t_0)| < \delta$ and $D_t(u(t)) < M_u$ for all $t \ge t_0$, then $|e(t)| \le r \le \varepsilon$ for all $t \ge t_0$. Proceed by contadiction. Suppose $|e(t)|$ leaves the ball of radius r. Since $|e(t)|$ varies continuously in t, there exists a smallest time T_2 such that $|e(T_2)| = r$. Likewise, there exists a largest time $T_1 < T_2$ such that $|e(T_1)| = \delta$.

With these choices of T_1 and T_2, it must be true that $\delta < |e(t)| < r \leq \varepsilon$ on the interval (T_1,T_2). On this same time interval, however, our choice of M_u in conjunction with (6) guarantees that

$$\dot{V}(e(t),t) < 0.$$

Consequently,

$$V(e(T_2), T_2) < V(e(T_1), T_1)$$

But

$$V(e(T_2), T_2) > \mu_{\min} r^2$$

and

$$V(e(T_1), T_1) < \mu_{\max}\delta^2 < \mu_{\min} r^2$$

which yields the desired contradiction.

CONCLUSIONS

The proof of the theorem is constructive, so that the bounds M_u and δ can actually be computed. This result quantifies the trade-off between the degree of stability of the family of linearizations, the rate at which the operating point is changing, and the maximal error between the desired operating point and the true state. The reader should note that the construction used here is employed to simplify the steps of the proof; less conservative bounds may easily be obtained.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the support of the Graduate School of the University of Wisconsin-Madison for funding this research.

REFERENCES

[1] B. R. Barmish and K. H. Wei, "Simultaneous stabilizability of single input - single output systems," in **Modelling, Identification, and Robust Control**, C. I. Byrnes and A. Lindquist, eds., Amsterdam: North Holland, 1986.

[2] M. Vidyasagar, **Nonlinear Systems Analysis**, Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1978.

[3] C.A. Desoer and M. Vidyasagar, **Feedback Systems: Input-Output Properties**, London: Academic Press, Inc., 1975.

[4] J. Barman, "Well-Posedness of Feedback Systems and Singular Perturbations," Ph.D. Dissertation, University of California, Berkeley, California, 1973.

[5] C.A. Desoer and Y.T. Wang, "The robust non-linear servomechanism problem," **International Journal of Control**, vol. 20, no. 5, pp. 803-828, 1979.

[6] A. Solomon and E.J. Davidson, "Design of controllers to solve the robust servomechanism problem for a class of nonlinear systems," **Proceedings of the 22nd Conference on Decision and Control**, San Antonio, Texas, pp. 335-341, Dec. 1983.

[7] C.A. Desoer and C-A Lin, "Tracking and disturbance rejection of MIMO nonlinear systems with PI controller," **IEEE Transactions on Automatic Control**, vol. AC-30, no. 9, pp. 861-867, Sept. 1985.

[8] E. A. Coddington and N. Levinson, **Theory of Ordinary Differential Equations**, New York: Academic Press, 1960.

A PARAMETRIZATION OF THE FIRST-ORDER OPTIMALITY CONDITIONS FOR A CLASS OF DISCRETE OPTIMAL CONTROL PROBLEMS

GERARD G. L. MEYER and LOUIS J. PODRAZIK
Electrical Engineering and Computer Science Department
The Johns Hopkins University
Baltimore, Maryland 21218

ABSTRACT

In this paper, we present a parametrization of the first-order optimality conditions for a general class of unconstrained discrete optimal control problems. Based upon this parametrization, we show that parallel evaluation of the gradient of the cost function with respect to the control can be efficiently performed provided that proper parameters are chosen. For a particular choice of parameters we easily recover the classical optimality conditions which involve the state and costate difference equations obtained through the Hamiltonian formulation of the problem. Moreover, we show that the parametrization is applicable to cases in which the cost function is not additively separable.

I. INTRODUCTION

Previous parallel approaches to the solution of optimal control problems [LAR73], [MEY73], [TRA80] have been devised without explicitly taking into consideration the computational environment. In particular, when the number of available processors is small in relation to the problem size, the above techniques simply fold the computations to fit the number of processors. More efficient parallel algorithms may be devised by considering the computational environment throughout the algorithm synthesis. Toward that end, we present in this paper a parametrization of the first-order optimality conditions for a class of unconstrained discrete optimal control problems. Based upon this parametrization, we show that parallel evaluation of the gradient of the cost function with respect to the control exhibits a high degree of parallelism provided that proper parameters are chosen as a function of the number of available processors. Furthermore, although presented in the context of unconstrained optimal control, our results for gradient computation are also applicable to constrained problems whenever the problem is solved using techniques which do not modify the cost function.

Our approach to parallel gradient computation is based upon a two-level parametrization of the first-order optimality conditions. The first level of parametrization consists of introducing non-singular parameter matrices whenever possible and then choosing the structure of these matrices to decrease the overall amount of computation necessary for gradient evaluation. The second level of parametrization consists of selecting a factorization of the inverse of the parameter matrices obtained from the first level to organize the computation to exhibit the parallelism required by the computing network. In the special case in which the cost function is additively separable, we show that our choice of parameters recovers the usual first-order optimality conditions obtained through the Hamiltonian formulation of the problem.

The degree of parallelism exhibited by the gradient computation technique presented in this paper varies as a function of P, the number of processors in the computer network. Our gradient computation approach is applicable whenever $1 \leq P \leq nN^{\frac{1}{2}}$, where n is the size of the system state vector, N is the number of stages in the control process and m, the size of the control, satisfies $m \leq n$.

An efficient technique for gradient evaluation using a single processor has been discussed previously by Polak [POL71, pp.66-69]. A direct implementation of this technique on P processors achieves linear speedup for P up to n; however, for $P > n^2$, the speedup is significantly reduced. In this paper, we present an approach to gradient computation that may be efficiently implemented on up to $nN^{\frac{1}{2}}$ processors and reduces to a direct parallel implementation of gradient evaluation as reported by Polak when $1 \leq P \leq n^2$. For $n^2 < P \leq nN^{\frac{1}{2}}$, we show that our approach achieves speedup greater than $\dfrac{P+n}{3/2+n}$.

A critical step in our approach involves the parallel solution of block linear recurrence systems. Parallel evaluation of m-th order linear recurrence systems has been extensively studied [KOG73], [MEY85] and [MEY86]. To solve first-order linear block recurrence systems in parallel, we use a parametrized blocked formulation of the approach presented in [MEY85],[MEY86].

The organization of this paper is as follows : in Section II, we state the unconstrained discrete control problem and give various forms of its associated cost function gradient. Section III presents the first level of parametrization and gives examples of parameter choices that decrease the overall amount of computation necessary for gradient evaluation. Based upon the parameters discussed in Section III, Section IV presents the second level of parametrization that deals with effectively organizing the computation for parallel evaluation. Finally, in Section V conclusions are presented.

II. PRELIMINARIES

We consider the following discrete optimal control problem:
Problem 1: Given an m-input, discrete, time-varying, non-anticipative, nonlinear system in which we are given the initial state, z_0, and

$$z_i = f_i(z_{i-1}, z_{i-2}, \ldots, z_0, u_i, u_{i-1}, \ldots, u_1), \quad i = 1,2,\ldots N, \tag{2.1}$$

where for $i = 0,1,\ldots,N$, z_i in E^n is the state of the system at time i and for $i = 1,2,\ldots,N$, u_i in E^m is the control at time i, find the mN control vector $u = (u_1^t, u_2^t, \ldots, u_N^t)^t$ that minimizes the performance index

$$J = J(z, u), \tag{2.2}$$

where z is the nN vector $z = (z_1^t, z_2^t, \ldots, z_N^t)^t$, and the functions J and $f_i(\ ,\ldots,\ )$ are continuously differentiable in all their arguments.

The quantity $\dfrac{dJ}{du}$ is basic for formulating necessary conditions of optimality for Problem 1 and plays a fundamental role in many iterative algorithms used to solve it. Let J_u be the $1 \times mN$ Jacobian matrix $\frac{\partial J}{\partial u}$ and and J_z be $1 \times nN$ Jacobian matrix $\frac{\partial J}{\partial z}$. A direct application of the chain rule yields

$$\frac{dJ}{du} = J_u + J_z \frac{\partial z}{\partial u}, \tag{2.3}$$

where the z_i's are determined in accordance with Eq. (2.1), and $\frac{\partial z}{\partial u}$ is the $nN \times mN$ block lower triangular Jacobian matrix that consists of N^2 $n \times m$ blocks $\left(\frac{\partial z}{\partial u}\right)_{ij} = \frac{\partial z_i}{\partial u_j}$ obtained by the chain rule for all i and j in $[1,2,\ldots,N]$ by

$$\frac{\partial z_i}{\partial u_j} = \begin{cases} 0 & \text{if } i < j \\ \dfrac{\partial f_i}{\partial u_i} & \text{if } i = j \\ \dfrac{\partial f_i}{\partial u_j} + \dfrac{\partial f_i}{\partial z} \dfrac{\partial z}{\partial u_j} & \text{if } i > j. \end{cases} \tag{2.4}$$

Eq. (2.4) shows that the matrix $\frac{\partial z}{\partial u}$ satisfies $F_z \frac{\partial z}{\partial u} = F_u$, where F_z is the $nN \times nN$ block lower triangular matrix that consists of N^2 $n \times n$ blocks $\left(F_z\right)_{ij}$ defined for all i and j in $[1,2,\ldots,N]$ by

$$\left(F_z\right)_{ij} = \begin{cases} 0 & \text{if } i < j \\ I & \text{if } i = j \\ -\dfrac{\partial f_i}{\partial z_j} & \text{if } i > j \end{cases} \tag{2.5}$$

and F_u is the $nN \times mN$ block lower triangular matrix that consists of N^2 $n \times m$ blocks $\left(F_u\right)_{ij}$ defined for all i and j in $[1,2,\ldots,N]$ by

$$\left(F_u\right)_{ij} = \begin{cases} 0 & \text{if } i < j \\ \dfrac{\partial f_i}{\partial u_j} & \text{if } i \geq j. \end{cases} \tag{2.6}$$

F_z is non-singular and thus we may rewrite Eq. (2.3) as

$$\frac{dJ}{du} = J_u + J_z F_z^{-1} F_u. \tag{2.7}$$

It follows that given u, J_z, J_u, F_z and F_u, the $mN \times 1$ gradient of J with respect to u, $g(u)$, may be obtained by introducing the nN vector $\lambda = (\lambda_1^t, \lambda_2^t, \ldots, \lambda_N^t)^t$, λ_i in E^n, that is

$$z_i = f_i(z_{i-1}, z_{i-2}, \ldots, z_0, u_i, u_{i-1}, \ldots, u_1), \quad i = 1,2,\ldots N \tag{2.8}$$

$$\lambda = J_z^t \tag{2.9}$$

$$g(u) = J_u^t + \left(F_z^{-1} F_u\right)^t \lambda. \tag{2.10}$$

Therefore, u^* is a solution of Problem 1 only if there exists z and λ such that Eqs. (2.8) and (2.9) are satisfied and $g(u^*) = 0$.

In the next section we present the first level of a parametrized approach for the computation of $g(u)$, give examples of parameter matrices that decrease the number of operations required to compute $g(u)$ and then examine the corresponding parallel implementation in a multiprocessor environment using P processors, where $1 \leq P \leq nN$. We assume N to be large with respect to m and n. To simplify the analysis, we consider an idealized model of parallel computation which consists of a global parallel memory, P parallel processors and a control unit. The model is simplified by making the following usual assumptions: (i) Each arithmetic operation takes the same amount of time, referred to as a step, (ii) There are no accessing conflicts in global memory; furthermore, all data is assumed to initially reside in global memory and (iii) There is no time required to access global memory.

III. FIRST LEVEL OF PARAMETRIZATION

The first level of parametrization consists of introducing non-singular matrices S_λ, S_u, T_u, S_z and T_z when possible, and then chosing the structure of these matrices to decrease the overall amount of computation necessary to evaluate $g(u)$.

Given a non-singular $nN \times nN$ matrix S_λ, we may rewrite Eqs. (2.9) and (2.10) as

$$S_\lambda^t \, \tilde{\lambda} = J_z^t \tag{3.1}$$

$$g(u) = J_u^t + \left(S_\lambda F_z^{-1} F_u\right)^t \tilde{\lambda}, \tag{3.2}$$

and thus the following algorithm computes $g(u)$.

```
Algorithm 1: Given $S_\lambda$, $z_0$ and $u$.
Step 1: Compute $z_i = f_i(z_{i-1}, \ldots, z_0, u_i, \ldots, u_1)$, $i = 1,2,\ldots N$.
Step 2: Compute $F_z$ and $F_u$ as given in Eqs. (2.5) and (2.6).
Step 3: Compute $\alpha = J_z^t$.
Step 4: Compute $\tilde{\lambda}$ such that $S_\lambda^t \tilde{\lambda} = \alpha$.
Step 5: Compute $\gamma = S_\lambda F_z^{-1} F_u$.
Step 6: Compute $\delta = \gamma^t \tilde{\lambda}$.
Step 7: Compute $\beta = J_u^t$.
Step 8: Compute $g(u) = \beta + \delta$.
```

If we choose S_λ to be the $nN \times nN$ identity matrix, Algorithm 1 reduces to the computation of $g(u)$ suggested by Eqs. (2.8), (2.9) and (2.10). To improve upon that choice, we need to determine a matrix S_λ such that the total number of operations required to solve the linear system $S_\lambda^t \tilde{\lambda} = \alpha$ in step 4 and compute the matrix product $S_\lambda F_z^{-1} F_u$ in step 5 is smaller than the number of operations required to compute the matrix product $F_z^{-1} F_u$. For example, if we choose $S_\lambda = F_z$, then step 4 of Algorithm 1 reduces to solving the block upper triangular linear system $F_z^t \tilde{\lambda} = J_z^t$ and the matrix γ in step 5 is simply F_u. Thus, the choice $S_\lambda = F_z$ results in a reduction of the overall gradient computational requirements by removing the $nN \times nN$ by $nN \times mN$ matrix product of step 5 and adding a

$nN \times nN$ by $nN \times 1$ matrix-vector product to step 4.

Note 1: If the cost function J is additively separable in z_i and u_i, that is

$$J(z,u) = \sum_{i=1}^{N} J_i(z_i, u_i), \tag{3.3}$$

and if $S_\lambda = F_z$, Algorithm 1 reduces to the usual approach to computing $g(u)$ based on state and costate recurrence equations produced by the Hamiltonian formulation of Problem 1. Thus the Hamiltonian formulation may be viewed as an efficient approach to gradient computation when J is separable.

Note 2: If the cost function J is additively separable, if the system dynamics are given by

$$z_i = f_i(z_{i-1}, u_i), \quad i = 1,2,\ldots N, \tag{3.4}$$

and if S_λ is chosen to be equal to F_z, Algorithm 1 reduces to Polak' s approach for gradient computation [POL71].

If the cost function J satisfies

$$J_u^t = Ru + c, \tag{3.5}$$

for some symmetric $mN \times mN$ matrix R and some mN vector c, we may use non-singular $mN \times mN$ parametrizing matrices S_u and T_u to rewrite Eq. (3.2) as

$$T_u^t \tilde{u} = u \tag{3.6a}$$

$$S_u \ g(u) = \left(S_u R T_u^t\right)\tilde{u} + S_u c + S_u \left(S_\lambda F_z^{-1} F_u\right)^t \tilde{\lambda} \tag{3.6b}$$

and thus the following algorithm computes $g(u)$, where the quantities $S_u R T_u^t$ and $S_u c$ may be precomputed offline.

Algorithm 2: Given S_λ, S_u, T_u, $S_u R T_u^t$, $S_u c$, z_0 and u.

Step 1: Compute $z_i = f_i(z_{i-1}, \ldots, z_0, u_i, \ldots, u_1)$, $i = 1,2,\ldots N$.

Step 2: Compute F_z and F_u as given in Eqs. (2.5) and (2.6).

Step 3: Compute $\alpha = J_z^t$.

Step 4: Compute $\tilde{\lambda}$ such that $S_\lambda^t \tilde{\lambda} = \alpha$.

Step 5: Compute $\gamma = S_\lambda F_z^{-1} F_u$.

Step 6: Compute $\delta = \gamma^t \tilde{\lambda}$.

Step 7: Compute $\tilde{u}$ such that $T_u^t \tilde{u} = u$.

Step 8: Compute $\tilde{\beta} = \left(S_u R T_u^t\right)\tilde{u} + S_u c$.

Step 9: Compute $g(u)$ such that $S_u \ g(u) = \tilde{\beta} + S_u \delta$.

The matrices S_λ, S_u and T_u are chosen to reduce the overall number of operations necessary to obtain $g(u)$. The matrix S_λ is involved in steps 4 and 5, the matrices S_u and T_u are involved in steps 7,8 and 9 and therefore S_u and T_u may be chosen independently of S_λ. We have already discussed the choice of S_λ in the context of Algorithm 1, and we need only consider the appropriate choices for S_u and T_u. The motivation for the choice of S_u and T_u is to transform the matrix $S_u R T_u^t$ into a sparse matrix while at the same time producing linear systems involving S_u and T_u^t which are easy to solve. Thus, if R is diagonal then the best choice for S_u and T_u is clearly the identity matrix. The following example shows one possible structure of R and the corresponding choice for S_u and T_u which reduces the overall number of operations required for gradient computation.

Example 1: Let R be the symmetric $mN \times mN$ matrix that consists of N^2 $m \times m$ non-zero blocks $(R)_{ij}$ defined for all i and j in $[1,2,\ldots,N]$ by $(R)_{ii} = W$, $(R)_{ij} = V_i \cdots V_{j+1} W$, $i > j$, and $(R)_{ij} = (R)_{ji}^t$, $i < j$, where W is a symmetric $m \times m$ matrix. Choose $S_u = T_u$ where S_u is the $mN \times mN$ block bidiagonal matrix that consists of N^2 $m \times m$ blocks $\left(S_u\right)_{ij}$ defined for all i and j in $[1,2,\ldots,N]$ by $\left(S_u\right)_{ii} = I$, $\left(S_u\right)_{ij} = -V_i$ if $i = j+1$ and $\left(S_u\right)_{ij} = 0$, otherwise. For this choice of T_u step 7 of Algorithm 2 reduces to computing the reverse block linear recurrence $\tilde{u}_i = V_{i+1}^t \tilde{u}_{i+1} + u_i$, $i = N-1, N-2, \ldots, 1$, where the final condition is $\tilde{u}_N = u_N$. Let $\tilde{\beta}$ be the mN vector $\tilde{\beta} = (\tilde{\beta}_1^t, \ldots, \tilde{\beta}_N^t)^t$, $\tilde{\beta}_i \in E^m$, $\tilde{c}$ be the mN vector $\tilde{c} = S_u c$ and the $m \times m$ blocks Λ_i be defined as $\Lambda_1 = W$ and $\Lambda_i = W - V_i W V_i^t$, $i = 2,3,\ldots,N$. Then computation of $\tilde{\beta}$ in step 8 reduces to computing $\tilde{\beta}_i = \Lambda_i \tilde{u}_i + \tilde{c}_i$, for $i = 1,2,\ldots,N$. The computation of $g(u)$ in step 9 reduces to computing the forward block linear recurrence $g_i(u) = V_i(g_{i-1}(u) - \delta_{i-1}) + \tilde{\beta}_i + \delta_i$, $i = 2,3,\ldots,N$, where the initial condition is $g_1(u) = \tilde{\beta}_1 + \delta_1$. Thus, for this particular choice of S_u and T_u the number of operations required by steps 7,8 and 9 of Algorithm 2 is $2N(3m^2+m) - (4m^2+m)$. This improves upon the number of operations $2N^2m^2 + Nm$ required by steps 7,8 and 9 when S_u and T_u are chosen as the identity matrix.

If the system is non-anticipative and linear, that is, if Eq. (2.8) may be written as

$$F_z z = F_u u + F_0 z_0 \tag{3.7}$$

where F_0 is the $nN \times n$ block matrix that consists of N $n \times n$ blocks $\left(F_0\right)_i$ defined by $\left(F_0\right)_i = \partial f_i / \partial z_0$ for all i in $[1,2,\ldots,N]$, F_z and F_u are defined in Eq. (2.5) and (2.6), and if the cost function J satisfies

$$J_z^t = Qz + d, \tag{3.8}$$

for some symmetric $nN \times nN$ matrix Q and some nN vector d, we may use non-singular parametrizing matrices S_z and T_z to rewrite Eqs. (3.1) and (3.2) as

$$F_z T_z^t \tilde{z} = F_u u + F_0 z_0 \tag{3.9}$$

$$S_z S_\lambda^t \tilde{\lambda} = \left(S_z Q T_z^t\right) \tilde{z} + S_z d, \tag{3.10}$$

and thus the following algorithm computes $g(u)$, where the quantities $S_z Q T_z^t$ and $S_z d$ may be precomputed offline.

Algorithm 3: Given S_λ, S_z, T_z, $S_z Q T_z^t$, $S_z d$, z_0 and u.

Step 1: Compute z such that $F_z z = F_u u + F_0 z_0$.

Step 2: Compute $\tilde{z}$ such that $T_z^t \tilde{z} = z$.

Step 3: Compute $\tilde{\alpha} = \left(S_z Q T_z^t\right) \tilde{z} + S_z d$.

Step 4: Compute τ such that $S_z \tau = \tilde{\alpha}$.
Step 5: Compute λ such that $S_\lambda^t \tilde{\lambda} = \tau$.
Step 6: Compute $\gamma = S_\lambda F_z^{-1} F_u$.
Step 7: Compute $\delta = \gamma^t \tilde{\lambda}$.
Step 8: Compute $\beta = J_u^t$.
Step 9: Compute $g(u) = \beta + \delta$.

The matrices S_λ, S_z and T_z are chosen to reduce the overall number of operations necessary to obtain $g(u)$. The matrix S_λ is involved in steps 5 and 6, the matrices S_z and T_z are involved in steps 2,3 and 4 and therefore S_z and T_z may be chosen independently of S_λ. The motivation for the choice of S_z and T_z is to transform the matrix $S_z QT_z^t$ into a sparse matrix while at the same time producing linear systems involving S_z and T_z^t which are easy to solve. This case is identical to that of choosing the parameter matrices S_u and T_u, except that the choice of S_z and T_z is dependent upon the structure of Q rather than R. Thus, the choice of S_z and T_z is a function of the same matrix structures as in the case of choosing S_u and T_u.

We now consider the parallelism exhibited by Algorithms 1,2 and 3. To simplify the discussion, we assume $m = n$ and the number of available processors P to be a multiple of n. Evaluation of the quantities J_u, J_z, F_z and F_u is a computation which inherently possesses a high degree of parallelism since each of the components can be evaluated independently of each other. Furthermore, the matrix operations $nN \times nN$ matrix-vector multiplication and nN vector addition exhibit linear speedup when the number of processors is in the range $1 \leq P \leq nN$. However, each of the previous algorithms also need to solve sparse linear systems. In most cases of interest, these systems are block triangular. We know that such systems may be solved using techniques which exhibit linear speedup when $1 \leq P \leq n$ and sublinear speedup when $P > n$, thus dominating the total number of parallel steps required for gradient computation. In the next section we investigate the solution of block triangular linear systems in the context of gradient computation and give a parallel procedure for their solution.

IV. SECOND LEVEL OF PARAMETRIZATION

The second level of parametrization consists of factoring the inverse of the parameter matrices obtained from the first level so as to organize the computation to exhibit the parallelism required by the given computing network. In addition, if the system dynamics are linear, then we also factor the matrix F_z^{-1}. Consider the case when all five parameter matrices S_λ, S_u, T_u, S_z and T_z can be introduced. Let $S_\lambda^{-t} = S_\lambda^1 \cdots S_\lambda^{j(S_z)}$, $S_u^{-1} = S_u^1 \cdots S_u^{j(S_u)}$, $T_u^{-t} = T_u^1 \cdots T_u^{j(T_u)}$, $S_z^{-1} = S_z^1 \cdots S_z^{j(S_z)}$, $T_z^{-t} = T_z^1 \cdots T_z^{j(T_z)}$ and $F_z^{-1} = F_z^1 \cdots F_z^{j(F_z)}$. Then $\tilde{z}$, $\tilde{\lambda}$, $\tilde{u}$ and $g(u)$ are computed from a series of matrix-vector multiplications that can each be performed in parallel by rewriting Eqs. (3.9), (3.10), (3.6a) and (3.6b) as

$$\tilde{z} = T_z^1 \cdots T_z^{j(T_z)} F_z^1 \cdots F_z^{j(F_z)} \left(F_u u + F_0 z_0 \right) \tag{4.1}$$

$$\tilde{\lambda} = S_\lambda^1 \cdots S_\lambda^{j(S_\lambda)} \left(S_z^1 \cdots S_z^{j(S_z)} \left(S_z QT_z^t \right) \tilde{z} + d \right) \tag{4.2}$$

$$\tilde{u} = T_u^{1} \cdots T_u^{j(T_u)} \, u \tag{4.3}$$

$$g(u) = S_u^{1} \cdots S_u^{j(S_u)} \left(S_u R T_u^t \right) \tilde{u} + c + \left(S_\lambda F_z^{-1} F_u \right)^t \tilde{\lambda}, \tag{4.4}$$

where the quantities $S_z Q T_z^t$ and $S_u R T_u^t$ are precomputed.

In order to demonstrate a matrix factorization that is chosen as a function of the number of processors so that each resulting matrix-vector product is computed efficiently in parallel we consider the special case in which the system dynamics are given by Eq. (3.4). In that case only the parameter matrix S_λ may be introduced. Choosing $S_\lambda = F_z$ results in S_λ being a block lower bidiagonal matrix and therefore step 4 of Algorithm 1 reduces to solving a reverse block linear recurrence system in parallel and step 5 reduces to $\gamma = F_u$.

Our approach to the parallel execution of step 4 is a modification of the parallel approach presented in [MEY86] and consists of choosing the factorization $S_\lambda^{-t} = S_\lambda^1 \cdots S_\lambda^{j(S_\lambda)}$ as a function of the number of processors and then executing step 4 as $\tilde{\lambda} = (S_\lambda^1 \cdots (S_\lambda^{j(S_\lambda)} \alpha)...)$. Instead of explicitly giving the matrix factorization, we give the procedure that computes $\tilde{\lambda}$ based upon our choice of factorization for S_λ^{-t}. If $p > n^2$, let $\Omega = \frac{N-1}{(P/n)^2-1)}$ and $\kappa = P/n$, otherwise let $\Omega = N-1$ and $\kappa = 1$. For ω in $[0,1,...,\Omega-1]$, define the index sets

$$r_0(\omega) = \{\kappa i + \omega(\kappa^2-1) : i = \kappa-1, \kappa-2, ..., 1\}$$

and

$$r_1(\omega) = \{\kappa i + Max(\omega+1, \omega(\kappa^2-1)) : i = \kappa, \kappa-1, ..., 1\}.$$

For presentation simplicity, we assume that Ω and P/n is an integer.

```
PROCEDURE $R(N, n, P, F_z, \alpha)$
 $\tilde{\lambda}_N := \alpha_N$
 FOR $\omega := \Omega - 1$ TO 0 DO
     FORALL $i \in r_0(\omega)$ DO IN PARALLEL $\Phi[i,i] := \frac{\partial f_i}{\partial z_{i-1}}$;
     FOR $i := 1$ TO $(\kappa-1)$ DO
        FORALL $j \in r_0(\omega)$ DO IN PARALLEL
           $\Phi[i+j, j] := \frac{\partial f_{i+j}}{\partial f_{i+j-1}} * \Phi[i+j-1, j]$;
        END FORALL
     END FOR
     FORALL $j \in r_0(\omega)$ DO IN PARALLEL $\tilde{\lambda}_j := \alpha_j$;
     FOR $i := 1$ TO $Max(1, \kappa-1)$ DO
        FORALL $j \in r_1(\omega)$ DO IN PARALLEL
           $\tilde{\lambda}_{j-i} := \frac{\partial f_{j-i+1}}{\partial z_{j-i}}^t * \tilde{\lambda}_{j-i+1} + \alpha_{j-i}$;
        END FORALL
     END FOR
     FORALL $j \in r_0(\omega)$ DO
        FOR $i := 0$ TO $\kappa-1$ DO IN PARALLEL
```

```
18.                 $\tilde{\lambda}_{j-i} := \Phi[j+1,j-i+1]^t * \tilde{\lambda}_{j+1} + \tilde{\lambda}_{j-i}$;
19.             END FOR
20.         END FORALL
21.     END FOR
22. END PROCEDURE
```

When $1 \le P \le n^2$, the index set $r_0(\omega)$ is empty and Procedure $R(N,n,P,F_z,\alpha)$ reduces to sequentially executing step 7 $N-1$ times, each execution using P processors. When $n^2 < P \le nN^{1/2}$, Procedure $R(N,n,P,F_z,\alpha)$ sequentially solves Ω reduced block recurrence systems in $\tilde{\lambda}_i$ of size $(P/n)^2$, each in parallel. Each reduced system is solved in three phases: the matrix coefficient phase consists of the execution of loops 4 and 5 and computes all products of the form $\frac{\partial f_{i+j}}{\partial z_{i+j-1}} \cdots \frac{\partial f_j}{\partial z_{j-1}}$ which are needed later during the solution updates; the second phase consists of the execution of steps 10 and 11 and computes $(P/n)^2$ partial solutions, in which the first P/n are the actual solutions; and finally, the third phase consists of the execution of loop 16 in which the matrix coefficients computed in the first phase are used to update the next P/n partial solutions at each iteration. We assign n processors to perform each of the P/n concurrent executions of steps 7,13 and 18. The complete solution to the block recurrence system is obtained after executing Ω iterations of loop 3. If Ω is not an integer, then we replace Ω by $\lceil \Omega \rceil$ and simply terminate the computation when λ_N is computed and if κ is not an integer, we replace κ by $\lfloor P/n \rfloor$.

If F_z or the choice of parameter matrices S_u or S_z requires that we solve a forward first-order block linear recurrence system in parallel, we use a blocked version of the first-order forward recurrence solver given in [MEY86]. Although similar to the technique used to solve reverse systems, we note that the matrix factorization used to solve forward systems is not simply the transpose of the factorization performed for reverse systems.

We now give the number of steps required to solve either a forward or reverse first-order block linear recurrence system.

Lemma 1: Given N, n and P such that $\frac{N-1}{(P/n)^2-1}$ is an integer, the number of parallel steps required to solve a block $n \times n$ first-order linear recurrence system of length N using P processors is

$$T(N,n,p) = \begin{cases} 2\left\lceil \frac{n}{p} \right\rceil n(N-1) & \text{if } 1 \le P \le n^2 \\ \frac{N-1}{P/n+1}(2n^2+3n) & \text{if } n^2 < P \le nN^{1/2}. \end{cases}$$

Using Procedure $R(N,n,P,F_z,\alpha)$ we now examine how the choice of $S_\lambda = F_z$ improves upon the choice $S_\lambda = I$ in the case of Algorithm 1.

Theorem 1: Given z_0, u and the system described by Eq. (3.4), the number of parallel steps required by Algorithm 1 to compute $g(u)$ using P processors, $1 \le P \le nN^{1/2}$, may be reduced by more than $\frac{n^3N^2}{P+n}$ steps when S_λ is chosen to be F_z instead of the identity matrix.

Finally, if F_z or the choice of parameter matrices S_λ, S_u, T_u, S_z or T_z requires that we solve a banded triangular linear system in parallel then we use the technique presented by Kogge and Stone [KOG73] to convert a banded system into a first-order system and then solve the resulting system using our parallel approach. Rather than repeat their presentation, we simply observe that the generalization leads to similar results for the parallel solution of banded triangular linear systems.

V. CONCLUSIONS

A two-level parametrization of the first-order optimality conditions for a general class of unconstrained discrete optimal control problems has been presented. The parametrization is applicable to cases in which the cost function is not additively separable and in the special case when the cost is separable, our choice of the first-level parameters recovers the usual optimality conditions obtained from the Hamiltonian formulation of the problem. Finally, we observe that our results for parallel gradient computation are also applicable to constrained versions of the problems considered in this paper whenever gradient based techniques which do not modify the cost function are used.

ACKNOWLEDGEMENT

This work was supported by the Air Force Office of Scientific Research under Contract AFOSR-85-0097.

REFERENCES

[KOG73] Kogge, P.M., and Stone, H.S., A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations, *IEEE Trans. Computers.,* Vol. C-22, No. 8, August 1973, pp. 786-793.

[LAR73] Larson, R.E. and Edison, T., Parallel Processing Algorithms for the Optimal Control of Nonlinear Dynamic Systems, *IEEE Trans. Computers.,* Vol. C-22, No. 8, August 1973, pp. 777-786.

[MEY73] Meyer, G.G.L., A Segmented Algorithm for Solving a Class of State Constrained Discrete Optimal Control Problems, *Decision and Control Theory Conference,* San Diego, California, 1973, pp. 73-79.

[MEY85] Meyer, G.G.L., and Podrazik, L.J., A Matrix Factorization Approach to the Parallel Solution of First-Order Linear Recurrences, *Proceedings of the 23-rd Annual Allerton Conference on Communication, Control and Computing,* October 2-4, 1985, pp. 243-250.

[MEY86] Meyer, G.G.L., and Podrazik, L.J., A Parallel First-Order Linear Recurrence Solver, *Proceedings of the 20-th Annual Conference Information Sciences & Systems,* Princeton, New Jersey, March 19-21, 1986.

[POL71] Polak, E., *Computational Methods in Optimization, A Unified Approach,* Academic Press, New York and London, 1971.

[TRA80] Travassos, R. and Kaufman, H., Parallel Algorithms for Solving Nonlinear Two-Point Boundary-Value Problems Which Arise in Optimal Control, *Journal of Optimization Theory and Applications,* Vol. 30, No. 1, January 1980, pp. 53-71.

ON LINE SEARCH TERMINATION CRITERIA FOR AND CONVERGENCE OF CONIC ALGORITHMS FOR MINIMIZATION

K.A. ARIYAWANSA
Department of Pure and Applied Mathematics
Washington State University
Pullman, Washington 99164-2930.

We consider the unconstrained minimization problem:

Given $F : X \subseteq \Re^n \rightarrow \Re$, $F \in C^2$, produce a sequence $\{x_k\} \subset X$ that converges to a local minimizer x_* of F.

(1)

Currently the two most popular and successful classes of algorithms for (1) are the quasi-Newton algorithms and the conjugate direction algorithms. Both classes are based on local quadratic approximations (lqa's) to F or equivalently on local affine scalings (las's) of X. These are local in the sense that they are employed in a neighborhood of the current iterate.

Davidon [2,3] proposed using what he called local conic approximations (lca's) to F that generalize lqa's. Equivalently, he proposed using local collinear scalings (lcs's) of X that generalize las's of X. Details may be found in Davidon [3] but we mention that it is possible to utilize the additional degrees of freedom found in lca's and lcs's to make the local approximants to F at the current iterate interpolate *both* the function value and the gradient of F at both the current and previous iterates. Thus more information is incorporated into local approximants than is possible with lqa's and las's, and the consequent hope is that the resulting algorithms would perform better than lqa and las based algorithms.

Sorensen [8] and Ariyawansa [1] develop *different* classes of collinear scaling algorithms as natural generalizations of quasi-Newton algorithms. Davidon [4] and Gourgeon and Nocedal [6] have proposed collinear scaling algorithms that generalize conjugate direction algorithms.

The analyses of algorithms presented in these papers are done under *idealized* conditions in the sense that they assume x_0 is *close* to x_* ([1],[8]), or F is conic ([3],[6]) or that line searches are *exact* ([6]).

The purpose of this note is to present preliminary results on our attempts to develop conic algorithms with *inexact* line searches and to theoretically validate the line search termination criteria proposed.

We now present the class of algorithms derived in [1] together with a set of line search termination criteria. In what follows $\|\cdot\|$ denotes l_2 norms, $F_k = F(x_k)$ and $F'_k = F'(x_k)$.

The class of algorithms

Step 0: (Input) $x_0 \in \Re^n$, $h_0 \in \Re^n$, $H_0 \in \Re^{n \times n}$, H_0 symmetric and positive definite, $0 < \theta_1 < 1$, $0 < \theta_2 < 1$, $0 < \theta_3 < 1$ and $0 < \theta_4 < 1$. Set $k = 0$.

Step 1: Set $d_k = -H_k F'_k$, $s_k(\lambda) = \lambda d_k$ and $x_k(\lambda) = x_k + s_k(\lambda)$, $\lambda > 0$, and search $\Phi(\lambda) = (F(x_k(\lambda))$ over $\lambda > 0$, until a $\lambda = \lambda_k \in \Delta_k(\theta_1, \theta_2)$ or a $\lambda = \lambda_k \in \Omega_k(\theta_3)$ is found. Here $\Delta_k(\theta_1, \theta_2) = \{\lambda : \lambda > 0, \Phi(0) - \Phi(\lambda) \geq -\theta_1 \lambda \Phi'(0), \Phi'(\lambda) \geq (1-\theta_2)[(\Phi(0) - \Phi(\lambda))/(-\lambda\Phi'(0)]^2 \Phi'(0)\}$ and $\Omega_k(\theta_3) = \{\lambda : \lambda > 0, \Phi(0) - \Phi(\lambda) = -\theta_3 \lambda \Phi'(0)\}$. Set $s_k = s_k(\lambda_k)$ and $x_{k+1} = x_k(\lambda_k)$.
Comment: If $1 + h_k^T d_k > 0$, then $\lambda = 1/(1 + h_k^T d_k)$ is used as the initial guess in this search.

Step 2: If either $\lambda_k \in \Delta_k(\theta_1, \theta_2)$ or if $\lambda_k \in \Omega_k(\theta_3)$
and $D_k = [(F_k - F_{k+1})^2 - (F'_{k+1})^T s_k (F'_k)^T s_k] \geq \theta_4 \|s_k\|^4$ then, set

$$\begin{aligned}
\rho_k &= \sqrt{D_k} \\
\gamma_k &= -(F'_k)^T s_k / ((F_k - F_{k+1}) + \rho_k) \\
y_k &= \gamma_k F'_{k+1} - (1/\gamma_k) F'_k \\
\tilde{v}_k &= \gamma_k s_k \\
r_k &= y_k + \frac{1-\gamma_k}{\gamma_k^2}[\gamma_k^2 F'_{k+1} - \frac{s_k^T F'_k}{s_k^T b_k} b_k], \qquad s_k^T b_k \neq 0 \\
h_{k+1} &= ((1-\gamma_k)/\gamma_k)(b_k/(s_k^T b_k)), \qquad s_k^T b_k \neq 0
\end{aligned}$$

$$\begin{aligned}
H_{k+1} = {} & (I - \frac{\tilde{v}_k r_k^T}{r_k^T \tilde{v}_k}) H_k (I - \frac{r_k \tilde{v}_k^T}{r_k^T \tilde{v}_k}) + \frac{\tilde{v}_k \tilde{v}_k^T}{r_k^T \tilde{v}_k} \\
& - (1 - \phi_k)(r_k^T H_k r_k)(\frac{\tilde{v}_k}{\tilde{v}_k^T r_k} - \frac{H_k r_k}{r_k^T H_k r_k})(\frac{\tilde{v}_k}{\tilde{v}_k^T r_k} - \frac{H_k r_k}{r_k^T H_k r_k})^T
\end{aligned}$$

where

$$\phi_k > 1/[1 - (r_k^T H_k r_k)(\tilde{v}_k H_k^{-1} \tilde{v}_k)/(\tilde{v}_k^T r_k)^2].$$

Comment: A possible choice of b_k is $b_k = F'_k$. Then, $r_k = y_k/\gamma_k$.
Else set

$$h_{k+1} = 0$$
$$H_{k+1} = H_k.$$

Set $k := k + 1$ and go to *Step 1.*

Note that the set $\Delta_k(\theta_1, \theta_2)$ is related to the set $\Delta_k^P(c_1, c_2) = \{\lambda : \lambda > 0, \Phi(0) - \Phi(\lambda) \geq -c_1\lambda\Phi'(0), \Phi'(\lambda) \geq c_2\Phi'(0), 0 < c_1 < c_2 < 1, c_1 < 1/2\}$ that corresponds to the line search termination criteria used by Powell [7]. However, unlike in the case of $\Delta_K^P(c_1, c_2)$ we have not been able to show that $\Delta_k(\theta_1, \theta_2)$ is not empty under reasonable conditions on F. Even though $\Delta_k^P(c_1, c_2)$ is not empty theoretically, in practice, a line search algorithm may find it difficult to find a $\lambda_k \in \Delta_k^P(c_1, c_2)$. For such conditions Gill, Murray, Saunders and Wright [5] suggest using $\lambda_k \in \Omega_k(\theta_3)$. Note also that if $\lambda_k \in \Delta_k(\theta_1, \theta_2)$ then $D_k > 0$ which is necessary to form a cupped conic approximation to F at x_{k+1}. However, if $\lambda_k \in \Omega_k(\theta_3)$ then it is not guaranteed that $D_k > 0$. The line search termination criteria and the criterion $D_k \geq \theta_4\|s_k\|^4$ used to decide to form a new conic approximant at x_{k+1} are motivated by these observations.

The convergence of the member of the class of algorithm with $b_k = F'_k$ and $\phi_k = 1$ for all k

Note that this member generalizes the BFGS method. We refer to this member as algorithm $\mathcal{A}$. It is possible to state the following theorem on the sequence $\{x_k\}$ generated by applying algorithm $\mathcal{A}$ to convex F. The somewhat lengthy proof of the theorem will be presented elsewhere.

Theorem 1: Suppose that F is convex on $\Re^n$, $\|F''(x)\| \geq m$ for some $m > 0$ for all $x \in \Re^n$, and that $\|F''(x_1) - F''(x_2)\| \leq M\|x_1 - x_2\|$, $M > 0$ for all $x_1, x_2 \in L(x_0) = \{x : F(x) \leq F(x_0)\}$. Let the sequence $\{x_k\}$ be generated by algorithm $\mathcal{A}$ applied to F. Then

$$\lim_{k \to \infty} \|F'(x_k)\| = 0 \quad \text{and} \tag{2}$$

$$\lim_{k \to \infty} x_k = x_*. \tag{3}$$

Further suppose that $0 < \theta_4 \leq m^2/8$. Then $\{x_k\}$ converges to x_* q-superlinearly.

Conclusion

We have presented an implementable class of collinear scaling algorithms for unconstrained minimization. A member of this class that extends the BFGS method does possess desirable properties similar to those shown for the BFGS method by Powell (1976). However, a major draw back is that in order to guarantee q-superlinear convergence a parameter in the algorithm must be set to a value which depends on F.

Acknowledgements

The research reported in this paper was partially supported by NSF grant DMS-8414460 and DOE grant DE-FG06-85ER25007.

References

[1]. Ariyawansa, K.A. (1985), "A class of collinear scaling algorithms for unconstrained minimization : Derivation and local analyses", Technical Report 85-1, Department of Pure and Applied Mathematics, Washington State University, Pullman, Washington.

[2]. Davidon, W.C. (1977), "Optimization by nonlinear scaling", in *Numerical Software — Needs and Availability*, ed., D. Jacobs, Academic Press, London, pp.377–383.

[3]. Davidon, W.C. (1980), "Conic approximations and collinear scalings for optimizers", *SIAM J. Numer. Anal.*, Vol. 17, pp.268–281.

[4]. Davidon, W.C. (1981), "Conjugate directions for conic functions", in *Nonlinear Optimization 1981*, ed., M.J.D. Powell, Academic Press, pp.23–28.

[5]. Gill, P.E., W. Murray, M.A. Saunders and M.H. Wright (1979), "Two step-length algorithms for numerical optimization", Technical Report SOL 79-25, Department of Operations Research, Stanford University, Stanford, California.

[6]. Gourgeon, H. and J. Nocedal (1985), "A conic algorithm for minimization", *SIAM J. Sci. Stat. Comput.*, Vol. 6, pp.253–267.

[7]. Powell, M.J.D. (1976), "Some global convergence properties of a variable metric algorithm for minimization without exact line searches", *Nonlinear Programming, SIAM-AMS Proceedings*, Vol. 9, Am. Math. Soc., Providence, Rhode Island, pp.53–72.

[8]. Sorensen, D.C. (1980), "The q-superlinear convergence of a collinear scaling algorithm for unconstrained optimization", *SIAM J. Numer. Anal.*, Vol. 19, pp.409–426.

Controlled Discrete Events Generated by Diffusion-Threshold Processes

CHARLES J. CONRAD
Mictron, Inc.
1624 Meijer Dr.
Troy, Michigan 48084

N. HARRIS MCCLAMROCH
Department of Aerospace Engineering
The University of Michigan
Ann Arbor, Michigan 48109

CHIH-PING LEE
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

A class of discrete event (or point) processes are introduced that arise from an underlying diffusion-threshold process. This approach permits the natural incorporation of control through the evolution of a controlled diffusion. The general formulation is indicated, and a specific class is described where particular control theoretic results have been obtained. This general viewpoint is consistent with many applications where the rate at which discrete events occur is influenced by an underlying dynamic process.

1. INTRODUCTION

In this paper, a new approach to the formulation of controlled stochastic discrete event processes is proposed. Included in this class of processes are controlled queues and controlled renewal processes. Since such stochastic processes have wide applicability as models in manufacuturing, operations research and computer science (inventory, scheduling, production lines, reliability, waiting lines, etc.) an attempt to incorporate explicit control mechanisms into such processes is of substantial intellectual and practical interest.

Our intention is not only to introduce control mechanisms such that discrete control parameters can be adjusted, but it is also to allow the incorporation of system adaptivity based on information feedback. Such control capability goes beyond the selection of stopping times, routing strategies, or other discrete control actions; in particular a mechanism is proposed which allows the introduction of a general class of feedback control actions which influence the rate of evolution of the discrete event process.

The basic idea proposed in the paper is that discrete events arise as a result of some causal mechanism which can, in many cases, be modeled by a continuous time controlled (stochastic) dynamical system. It is a natural view that discrete events occur at times when the solution of the underlying dynamical system satisfies a specified condition. Control mechanisms can be introduced into the underlying continuous time dynamical system so that the control directly affects the evolution of the underlying dynamics and indirectly affects the evolution rate of the discrete event process. In addition, the control action in the underlying dynamical system may depend on feedback information consisting of the history of the discrete event process and/or the history of the underlying dynamical system. That is, feedback control may result in closed loop coupling between the discrete event process and the underlying controlled dynamics.

2. THE DIFFUSION-THRESHOLD PROCESS

Let x_t denote the solution to the controlled stochastic differential equation

$$dx_t = b(x_t,u_t)\,dt + \sigma(x_t,u_t)\,dw_t \tag{1}$$

corresponding to $x_0 = X_0$. Define the stopping time $T_1 > 0$ by

$$T_1 = \inf\{\, t > 0 : g(x_t) = 0 \,\} \tag{2}$$

where $b : R^2 \rightarrow R^1$ is the drift function, $\sigma : R^2 \rightarrow R^1$ is the diffusion function, and $g : R^1 \rightarrow R^1$ is the stopping function; here w_t is a standard Brownian motion. At time T_1, the diffusion process is reinitialized to $x_{T_1+} = X_1$ and x_t continues to evolve according to equation (1) until the stopping condition is again satisfied, defining $T_2 > T_1$. Continuing in this way, the process x_t, $t \geq 0$, and $\{T_i,\ i = 0, 1, 2, \ldots\}$ are successively defined, with $T_0 = 0$. Thus, x_t, $t \geq 0$, is a controlled diffusion [5,6], satisfying equation (1) except at T_i, $i = 0, 1, 2, \ldots$, where it may be discontinuous; and $\{T_i,\ i = 0, 1, 2, \ldots\}$ are defined recursively by

$$T_i = \inf\{\, t > T_{i-1} : g(x_t) = 0 \,\} \tag{3}$$

Assume the reinitialization sequence $\{X_i,\ i = 0, 1, 2, \ldots\}$ is a set of independent random variables.

We refer to the random process x_t, $t \geq 0$, as a diffusion-threshold process, to $\{T_i, i = 0, 1, 2, \ldots\}$ as the event times and to u_t, $t \geq 0$, as the control process. We also define a counting process N_t, for $t \geq 0$, such that N_t is piecewise constant with unit jumps at the event times T_i, $i = 0, 1, 2, \ldots$, initialized by $N_0 = 0$. Note that sample paths of x_t, $t \geq 0$, are piecewise continuous (left continuous) with jumps at the event times defined by the reinitialization sequence; an example sample path is shown in Figure 1.

If suitable assumptions are made about the control then $\{T_i, i = 1, 2, \ldots\}$ are controlled stopping times for the diffusion and they define a renewal process that specifies the times at which jumps in the counting process occur. Although the interevent times $\{T_i - T_{i-1}, i = 1, 2, \ldots\}$ may be independent their distribution is not exponential; hence the interevent times are not memoryless.

Extensions to the case where the counting process is vector valued, with each scalar process evolving according to an underlying continuous time dynamical system can be made. The details are complicated but should be conceptually clear.

Control action enters the diffusion-threshold model through the drift and diffusion coefficients, so that the control affects the intensity of the discrete event process. The control may be deterministic (open loop) or stochastic (closed loop). In the latter case, the control may be a stochastic process that is conditionally dependent on the event history and the history of the counting process [8].

We also mention an additional important feature of the above diffusion-threshold model for controlled discrete event processes. Since there is an underlying diffusion-threshold process explicit in the model, the control may depend on the history of that process. This allows incorporation of feedback from the diffusion-threshold process that characterizes the underlying dynamics that give rise to discrete event changes; this feature of the model has been found to be extremely important in certain applications [3,4].

The use of a diffusion-threshold model to capture the underlying dynamics of a discrete event process provides a view different than that which is usually found in the literature. In particular, the usual viewpoint of discrete event (point) processes have been classified as moment oriented or intensity oriented [1]. Although the diffusion-threshold approach is compatible with these other views, it also offers a different viewpoint, since the model allows for the existence and utilization of information structures not normally considered in the moment or intensity formulations. Of course, this additional information exacts a cost in terms of added complexity and computational difficulty. The diffusion-threshold model also provides a closer contact with the physics of event generation than is conveyed by either the moment or intensity points of view.

3. A SPECIFIC CLASS OF DIFFUSION-THRESHOLD PROCESSES

A class of controlled discrete event processes, defined from a diffusion-threshold process, has been studied in [3,4] where some specific control theoretic results have been obtained. This specific class is characterized by the assumption that the drift function b and the diffusion function σ are independent of the diffusion-threshold process x_t but may depend on the control process u_t; the reinitialization sequence is $X_i = 0$, $i = 0, 1, 2, \ldots$; and the stopping condition is defined by a threshold crossing, i.e. $g(x) = x - A$. Assume the control u_t is constant during the time between events and assume that $A > 0$, $b(u) > 0$, $\sigma(u) > 0$ where the control dependence is explicitly indicated. An analytic expression for the probability density of the time between events $\tau_i = T_i - T_{i-1}$ can be obtained as an explicit function of the control. In particular, the density for τ_i has been derived in [3,4] as

$$q_i(t;u) = \begin{cases} \dfrac{A}{\sqrt{2\pi}\ \sigma(u)\ t^{3/2}} \exp\left\{ \dfrac{-(A-b(u)t)^2}{2\ \sigma(u)^2\ t} \right\}, & t > 0 \\ 0, & t \leq 0. \end{cases} \tag{4}$$

where $u_t = u$, $T_{i-1} < t < T_i$.

This probability density is referred to as inverse gaussian and has been previously studied in [2,9]. Under the stated assumptions, a threshold crossing occurs almost surely, i.e. $P[\tau_i < \infty] = 1$, $i = 1, 2, \ldots$ The central moments are easily computed; we give only the mean and variance as

$$E[\tau_i] = A\, b(u)^{-1} \tag{5}$$

$$\mathrm{Var}[\tau_i] = A\, \sigma(u)^2\, b(u)^{-3}. \tag{6}$$

Thus the mean time between events is equal to the "distance" A divided by the positive drift rate b.

Note that this distribution is explicitly parameterized by the value of the control action through the drift and diffusion coefficients. It is this explicit parameterization that provides a clear means for dealing with controlled discrete event processes, for this class.

The above expression for the interevent density also holds if the control is constant between events, but with random values that are conditionally dependent on the event history and the history of the counting process. However, the interevent times may no longer be a renewal process.

4. COMPARISON WITH THE INTENSITY CHARACTERIZATION

Our formulation of controlled discrete event processes is in contrast with the developments in [1,8] where control mechanisms are abstractly formulated as influencing the intensity of the counting process. Here the control mechanism is explicitly formulated through the underlying diffusion-threshold process.

For the specific case described in Section 3 it is possible to characterize the discrete event process in terms of an intensity that can be explicitly computed using the inverse-gaussian density. Thus it is possible to compare the diffusion-threshold characterization and the intensity characterization.

For our purposes, the history will be the event time record, denoted by $\{ F_t, t \geq 0 \}$; this is the usual history considered. Then, heuristically, the intensity λ_t of the discrete event process is defined by

$$\lambda_t \, dt = P[\, dN_t = 1 \mid F_t \,] \tag{7}$$

where $dN_t = 1$ denotes a unit change in the counting process.

The intensity λ_t is necessarily a stochastic process. Howevever, it is easily characterized in terms of the inverse gaussian density. The intensity λ_t, $T_{i-1} < t < T_i$, given T_{i-1} and $u_t = u_{i-1}$, $T_{i-1} < t < T_i$ is

$$\lambda_t = \frac{q_i(t - T_{i-1}; u_{i-1})}{\int_{t-T_{i-1}}^{\infty} q_i(\eta\ ; u_{i-1})\, d\eta}, \quad T_{i-1} < t < T_i. \tag{8}$$

The key feature of the intensity is that it is a stochastic process with sample paths that are piecewise continuous with discontinuities at the event times; its functional dependence on the control is indicated in equation (8). It can also be viewed as a regnerative process. An example sample path for the intensity is shown in Figure 2.

APPLICATIONS OF DIFFUSION-THRESHOLD PROCESSES

It is important to make the case that diffusion-threshold models for controlled discrete event processes do represent, in a natural way, many physical situations. Our formulation has the advantage that it imposes a proper perspective in terms of model development; one simply examines the physical causal mechanisms that give rise to discrete events as a part of the modeling effort.

This approach has been effectively used in [3,4] to model failures in drilling operations, where drill failure is assumed to be caused by drill wear reaching a fixed threshold. In this case, the tool wear is the diffusion-threshold process, the cutting speed is the control variable which is assumed to influence the tool wear according to a simple diffusion with drift as indicated in Section 3. The tool wear is reinitialized to zero when a failure event occurs, corresponding to replacement of a new tool. The model in Section 3 was used to formulate and solve a stochastic optimal (feedback) control problem corresponding to economic selection of the cutting speed. The results in [3,4] also incorporate several additional features, including the notion of discrete parts and tool replacement before failure; details are given in [3,4].

In [3] diffusion-threshold models have been developed for failure control of multi-tooled machines by selection of cutting speed, for failure control of serial transfer lines by selection of operating speeds of individual machines, and for productivity control through supervision.

A diffusion-threshold process has also been used to model the service times in a M/IG/1 queue, so that the service time distribution is inverse gaussian and depends on a control "service rate". Our formulation in Section 3 is the basis for formulation and solution of the (open loop) optimal control problem of minimizing the weighted sum of sojourn time and control effort. Preliminary results are given in [7].

These efforts represent initial attempts to exploit the proposed framework for controlled discrete event processes. It is expected that this proposed framework can provide a basis for examination of many additional problems in manufacturing, operations research, and computer science.

REFERENCES

[1] Bremaud, P., *Point Processes and Queues*, Springer-Verlag, New York, 1981.

[2] Chhikara, R. S. and Folks, J. L., "The Inverse Gaussian Distribution as a Life Time Model," Technometrics, 19, 1977, 461-468.

[3] Conrad, C. J., *Stochastic Modeling and Control of Some Problems in Manufacturing*, Ph.D. dissertation, The University of Michigan, July, 1986.

[4] Conrad, C. J. and McClamroch, N. H., "The Drilling Problem: A Stochastic Modeling and Control Example in Manufacturing," submitted to IEEE Transactions on Automatic Control.

[5] Fleming, W. H. and Rishel, R. W., *Deterministic and Stochastic OptimalContol*, Springer-Verlag, New York, 1975.

[6] Krylov, N. V., *Controlled Diffusion Processes*, Springer-Verlag, New York, 1980.

[7] Lee, C. P., "Control of Queues with Service Times Modelled by Diffusion Threshold Processes," Department of Electrical Engineering and Computer Science, September, The University of Michigan, Ann Arbor, September, 1986.

[8] Snyder, D. L., *Random Point Processes*, John Wiley, New York, 1975.

[9] Tweedie, M. C. K., "Statistical Properties of Inverse Gaussian Distributions, I", Ann. Math. Stat., 28, 1957, 362-377.

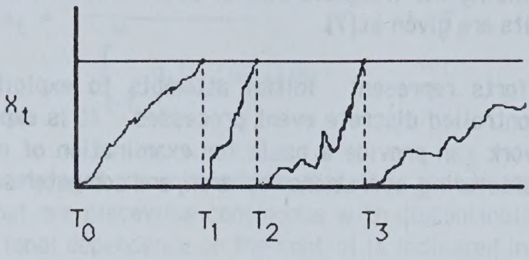


Fig. 1

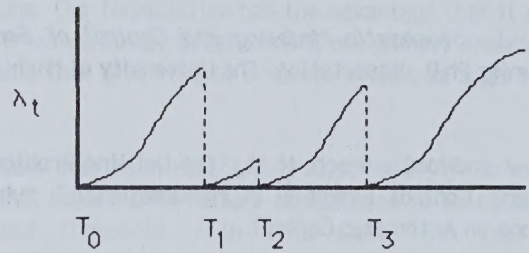


Fig. 2

STABILIZING CONTROL DESIGN FOR A CLASS OF STOCHASTIC NONLINEAR SYSTEMS

ENGIN YAZ
Electrical Engineering Department, University of Arkansas, Fayetteville
AR 72701

ABSTRACT

Two novel controllers are proposed for the purpose of stabilizing a large class of discrete-time stochastic nonlinear systems. The controllers are of constant feedback type and under the given conditions stabilize these systems in both mean square and almost sure senses.

1. INTRODUCTION

Due to their practical importance in control engineering, control and stability of several different classes of discrete-time stochastic nonlinear systems have been investigated. Both frequency domain (e.g. [1]) and time domain conditions ([2] and [3]) have been presented for guaranteeing stability. These conditions can effectively be used as starting points in designing stabilizing controllers for such systems. The special class of systems which we are interested in, have an additive vector nonlinearity which contains the state, input, and an independent noise vector as follows:

$$x_{k+1} = Ax_k + Bu_k + f(x_k, u_k, w_k) \tag{1}$$

where $x_k \varepsilon R^n$ is the state, $u_k \varepsilon R^m$ is the input, and $w_k \varepsilon R^p$ is a zero mean stationary, white noise sequence. We assume that x_o is deterministic and

$$E\{f(x_k, u_k, w_k)\} = 0 \tag{2}$$

for all x_k and u_k, where E{.} denotes the expectation operator. The second moment of f is given by

$$E\{f(x_k, u_k, w_k)f^T(x_k, u_k, w_k)\} = T^o + \sum_{i=1}^{n^1} T^i(\tfrac{1}{2}x_k^T W^i x_k + \tfrac{1}{2}u_k^T M^i u_k) \tag{3}$$

where $n^1 = n(n+1)/2$, and T^o, T^i, W^i, and M^i are all nonnegative definite, symmetric matrices. In [4], the problem of finding the optimal controller which minimizes a finite horizon quadratic performance index subject to a nonstationary version of system (1), is solved. As pointed in that work, the above system model is general enough to include common cases such as:

1. Linear system, additive noise: $f(x_k, u_k, w_k) = Cw_k$.
2. Linear system, multiplicative noise: $f(x_k, u_k, w_k^1, w_k^2) = w_k^1 Cx_k + w_k^2 Du_k$ where w_k^1 and w_k^2 are zero mean and independent scalar noise sequences.
3. Norm dependent noises: $f(x_k, u_k, w_k^1, w_k^2) = ||x_k||Cw_k^1 + ||u_k||Dw_k^2$
4. Noise vector dependent on the sign of a nonlinear function of x_k and u_k=

   $f(x_k, u_k, w_k) = \mathrm{sgn}[\ (x_k, u_k)]Cw_k$

In the following sections, we will see two different constant feedback controllers which render this class of nonlinear stochastic systems stable.

2. FINITE HORIZON STOCHASTIC CONTROLLER

Motivation behind using this controller is given in [5] and [6]. The control action is of a constant feedback type as follows:

$$u_k^1 = -\left(R(S_{N-1}) + B^T S_{N-1} B\right)^{-1} B^T S_{N-1} A x_k$$
$$\triangleq K(S_{N-1}) x_k \qquad (4)$$

where R(.) is defined as

$$R(X) = R + \tfrac{1}{2} \sum_{i=1}^{n_1} \mathrm{tr}[XT^i] M^i \qquad (5)$$

with $R=R^T>0$ (R is symmetric and positive definite) and tr[.] denotes the trace of a matrix. T^i and M^i are introduced in (3). S_{N-1} is found by (N-1)-fold application of the transformation Φ on F, a positive definite and symmetric matrix, or

$$S_{N-1} = \Phi^{N-1}(F) \qquad (6)$$

where

$$\Phi(X) = A^T XA - A^T XB\left(R(X) + B^T XB\right)^{-1} B^T XA + Q(X) \qquad (7)$$

with Q(.) given by

$$Q(X) = Q + \tfrac{1}{2} \sum_{i=1}^{n_1} \mathrm{tr}[XT^i] W^i \qquad (8)$$

where $Q=Q^T>0$, and W^i is given by (3). The positive integer N is called the horizon length, and as can be illustrated with simulations, determine the strength of the control action, and, therefore, the transient response of the closed loop system. The matrices F, Q, and R will be called the weighting matrices.

We will need the following auxiliary results to prove our stability result.

LEMMA 1. All finite-time solutions S_N of (6) are positive definite and symmetric.

PROOF. The initial condition is $S_0=F=F^T>0$. Letting $S_N>0$ and using (7) and matrix inversion lemma, it can be shown that $S_{N+1}>0$. An inductive reasoning implies positive definiteness. Symmetry follows from the symmetry of weighting matrices and the form of (7).

LEMMA 2. $\Phi(.)$ given by (7) is a monotonic transformation.

PROOF. Taking two positive definite matrices X and Y such that $X \geq Y$, it is possible to show, using the partial ordering results for matrices in [7], that $\Phi(X) \geq \Phi(Y)$.

Using these two lemmata, the stabilizing property of u_k^1 in (4) for system (1) can be proved to be:

THEOREM 1. Finite horizon stochastic controller u_k^1 given by (4) is stabilizing for all $N \geq 1$, for system (1) in the sense that, the closed loop system state is both mean square and almost sure bounded, if

$$F \geq \Phi(F) \tag{9}$$

PROOF. The proof is by the discrete stochastic version of the second method of Lyapunov in [2] and [5]. Taking

$$V_k^1 = x_k^T S_{N-1} x_k \tag{10}$$

as the Lyapunov function candidate, calculating the difference

$$E\{V_{k+1}^1 / x_k, \ldots, x_0\} - \frac{1}{\eta} V_k^1 \leq c \tag{11}$$

where both $\eta > 1$ and c are real constants. By the results in the above-cited works, the theorem follows.

If T_0 in (3) is identically zero, which means that there is no separate additive white noise effect on the system, the following result holds:

COROLLARY 1. Let $T_0 \equiv 0 \epsilon R^{nxn}$. Then under the same conditions as of Theorem 1, u_k^1 in (4) stabilizes system (1) in both almost sure and mean square asymptotic senses.

PROOF. Since $T_0 \equiv 0$ in this case, inequality (11) in the proof of Theorem 1, becomes

$$E\{V_{k+1} / x_k, \ldots, x_0\} - \frac{1}{\eta} V_k < 0 \tag{12}$$

which by [2] implies the result.

The inequality (9) has to be verified for positive definite F, Q, and R before u_k^1 is calculated. In order to justify a search for such matrices, their existence has to be ascertained. This can be done via the following result:

THEOREM 2. Let B be of full rank (without any loss of generality). If the following matrix inequality holds

$$I_n - A^T A + A^T B(B^T B + \tfrac{1}{2} \sum_{i=1}^{n^1} tr[T^i] M^i)^{-1} B^T A - \tfrac{1}{2} \sum_{i=1}^{n^1} tr[T^i] W^i > 0 \tag{13}$$

then it is possible to find weighting matrices so that (9) is satisfied.

PROOF. Since the matrix on the left hand side is positive definite, it is possible to find scalars q>0 and r>0 such that

$$I_n \geq \Phi(I_n) \tag{14}$$

where qI_n and rI_n are used instead of Q and R in (14).

Once condition (13) which does not depend on weighting matrices, is satisfied then, one can assume small values for q and r so that (14) is valid. The control u_k^1 designed based on these values, is guaranteed to be stabilizing for system (1).

3. FINITE HORIZON CERTAINTY EQUIVALENCE CONTROLLER

This controller is motivated by certainty equivalent results in [8] and [9]. We consider the following deterministic equivalent or certainty equivalent of system (1):

$$Z_{k+1} = AZ_k + Bu_k \qquad (15)$$

where the nonlinear term containing the random component is replaced by its expected value. For this deterministic system, we define the moving horizon cost criterion

$$J = Z_{k+N}^T FZ_{k+N} + \sum_{i=k}^{k+N-1} (Z_i^T QZ_i + u_i^T Ru_i) \qquad (16)$$

F, Q, and R being symmetric, positive definite matrices. Minimizing (16) subject to system constraint (15) leads to

$$u_k^0 = -(R+B^T P_{N-1} B)^{-1} B^T P_{N-1} A Z_k$$

$$= G(P_{N-1})Z_k \qquad (17)$$

where P_{N-1} is obtained by (N-1) - fold application of Γ on F as

$$P_{N-1} = \Gamma^{N-1}(F) \qquad (18)$$

with

$$\Gamma(X) = A^T XA - A^T XB(R+B^T XB)^{-1} B^T XA + Q \qquad (19)$$

The following theorem states the results in the case where the gain $G(P_{N-1})$ in (17), which is found using the linear deterministic equivalent system description, is used as a feedback of the states of system (1) as

$$u_k^2 = G(P_{N-1})x_k \qquad (20)$$

to control system (1).

THEOREM 3. Moving horizon certainty equivalent control in (20), when used in system (1), stabilizes it in the sense of rendering the closed loop system state mean square and almost surely bounded, if the following inequalities are valid

$$F \geq \Gamma(F) \qquad (21)$$

$$Q > \tfrac{1}{2} \sum_{i=1}^{n^1} tr[T^i F]W^i \qquad (22)$$

and

$$R > \tfrac{1}{2} \sum_{i=1}^{n^1} tr[T^i F]M^i \qquad (23)$$

PROOF. We know from [5] that the solutions of (18) are all positive definite, symmetric and also monotonically decreasing if (9) holds. Using the controller (20) in system (1), and trying the Lyapunov function candidate

$$V_k^2 = x_k^T P_{N-1} x_k \qquad (24)$$

we obtain

$$E\{V_{k+1}^2/x_k,\ldots,x_0\} - V_k^2 = x_k^T\{[A+BG(P_{N-1})]^T P_{N-1}[A+BG(P_{N-1})]-P_{N-1}]x_k + E\{f^T P_{N-1} f\} \qquad (25)$$

By using equations (1)-(3) and (19), inequalities (20)-(22), the interchangeability of trace and expectation operations, and the monotonicity property of P_j, it is possible to show that (11) holds with V_k^1 replaced by V_k^2. Therefore, the result follows.

Since inequality (21) is the stabilizability condition given in [5] for deterministic systems and (22) and (23) determine the upper bounds on the second order noise characteristics in terms of the weighting matrices in (16), we can infer from these two facts that if the noise powers in f, are small enough, then the stability robustness of the design based on the deterministic equivalent system will be sufficient to accommodate the noise effects.

The following result pertains to the case of no additive noise and therefore stronger.

COROLLARY 2. Under the same conditions as in Theorem 3, asymptotic stability in both senses of almost sure and mean square can be guaranteed if $T_0 \equiv 0$.

PROOF. Proof follows from the proof of Theorem 3. Since $T_0 \equiv 0$, we have (12) with V^1_k replaced by V^2_k, so the result follows.

4. CONCLUSION

Two stabilizing controller designs are presented for a class of discrete-time nonlinear stochastic systems. In the presence of additive noise, mean square and almost sure boundedness of the closed loop state vector are shown. In its absence, the system can be made asymptotically stable in both senses of mean square and almost sure stability. We have assumed that the initial state vector x_0 is deterministic but it can also be taken as an independent random vector.

REFERENCES

[1] M.K.P. Mishra and A.K. Mahalanabis, "On the stability of discrete nonlinear feedback systems with state dependent noise," Int. J. Syst. Science, vol. 6, pp. 479-490, 1975.

[2] R.G. Agniel and E.I. Jury, "Almost sure boundedness of randomly sampled systems," SIAM J. Control, vol. 9, pp. 372-384, 1971.

[3] E. Yaz and N. Yildizbayrak, "Robustness of feedback-stabilized systems in the presence of nonlinear and random perturbations," Int. J. Control, vol. 41, pp. 345-353, 1985.

[4] D.H. Jacobson, "A general result in stochastic optimal control of nonlinear discrete-time systems with quadratic performance criteria," J. Math. Anal. Applications, vol. 47, pp. 153-161, 1974.

[5] E. Yaz, "Stabilization of deterministic and stochastic-parameter discrete systems," Int. J. Control, vol. 42, pp. 33-41, 1985.

[6] E. Yaz, "Constant feedback stabilization of discrete-time systems with random coefficients," Int. J. Syst. Science, vol. 17, pp. 819-827, 1986.

[7] R. Bellman, Introduction to Matrix Analysis, McGraw-Hill, New York, 1970.

[8] E. Yaz, "Certainty equivalent control of stochastic systems: stability property," IEEE Trans. Autom. Control, vol. AC-31, pp. 178-180, 1986.

[9] E. Yaz, "Further results on the stabilizing property of certainty equivalent controllers," IEEE Trans. Autom. Control, vol. AC-31, pp. 586-587, 1986.

COMPENSATOR DESIGN FOR STOCHASTIC-PARAMETER SYSTEMS WITH INCOMPLETE STATE INFORMATION

ENGIN YAZ
Electrical Engineering Department, University of Arkansas, Fayetteville
AR 72701

SUMMARY

The problem of stabilizing a discrete stochastic-parameter system with incomplete state information using a time-invariant compensation scheme is addressed. The system and measurement equations are given by:

$$x_{k+1} = A_k(\omega)x_k + B_k(\omega)u_k \tag{1}$$

and

$$y_k = C_k(\omega)x_k \tag{2}$$

where $x_k \varepsilon R^n$ is the state, $u_k \varepsilon R^m$ is the input, and $y_k \varepsilon R^p$ is the output vector. Let x_o be deterministic for the time being. The coefficient matrices are of appropriate dimensions and have elements which are stationary, white, random sequences which are independent from each other.

Several physical systems which can be modelled in this fashion can be found in [1]. Stabilizability of system (1) by various different controllers, has been examined in the works [1] - [3], in the case of complete state information, and it has been pointed out that unlike white additive noise, stochastic parameters can destabilize the system. Stochastic state estimators are also proposed in [1].

In this work, we will be employing an observer-controller combination which has constant observer and feedback control gains as follows:

$$\hat{x}_{k+1} = \bar{A}\hat{x}_k + \bar{B}u_k + K^o(y_k - \bar{C}\hat{x}_k) \tag{3}$$

$$u_k = K^c\hat{x}_k \tag{4}$$

where overbar indicates the expected value of the random quantity, $\hat{x}_k$ is the vector of state estimates, and K^o and K^c are constant observer and control gain matrices, respectively, which are yet to be determined. So our objective is to find as large a class of time-invariant compensation schemes as possible to stochastically stabilize system (1), using available measurements (2).

By augmenting the stochastic observer equation (3) to the system equation (1) and separating the deterministic part which is given by the known means of the coefficient matrices from the zero mean stochastic part, the compensated system equation is

$$\xi_{k+1} = [\phi + \tilde{\phi}_k(\omega)]\xi_k \tag{5}$$

where

$$\phi = \begin{bmatrix} \bar{A} & \bar{B}K^c \\ K^o\bar{C} & \bar{A} - K^o\bar{C} + \bar{B}K^c \end{bmatrix} \tag{6}$$

and

$$\tilde{\phi}_k(\omega) = \begin{bmatrix} \tilde{A}_k(\omega) & \tilde{B}_k(\omega)K^c \\ K^o\tilde{C}_k(\omega) & 0 \end{bmatrix} \tag{7}$$

with $\tilde{A}_k(\omega)$ defined as

$$\tilde{A}_k(\omega) = A_k(\omega) - \bar{A} \tag{8}$$

and similarly for $\tilde{B}_k(\omega)$ and $\tilde{C}_k(\omega)$. We will base our designs on the deterministic part of system (5) and then we will check whether the degree of relative stability obtained by such a design can accommodate the stochastic part. The main result is given by the following theorem:

Theorem: Let $(\bar{A},\bar{B})$ be stabilizable and $(\bar{A},\bar{C})$ be detectable, and choose the gain matrices K^o and K^c such that $\bar{A} + \bar{B}K^c$ and $\bar{A} - K^o\bar{C}$ are both asymptotically stable. Let $P \varepsilon R^{2n \times 2n}$, which can be partitioned as

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix} \tag{9}$$

where each submatrix is square of dimension n, be the solution to

$$\phi^T P \phi - P = -I \tag{10}$$

then the compensated system (5) is asymptotically stable with probability one and in mean square sense if

$$\rho\{\overline{\tilde{A}^T P_{11} \tilde{A}} + \overline{\tilde{C}^T K^{oT} P_{22} K^o \tilde{C}}\} < 1 \tag{11}$$

and

$$\rho\{K^{cT} \overline{\tilde{B}^T P_{11} \tilde{B}} K^c\} < 1 \tag{12}$$

where $\rho\{.\}$ denotes the spectral radius of matrix in question.

The generalization of this formulation to stochastic parameter systems involving also additive noise terms in the state and measurement equations is straightforward, but this extention would be at the expense of losing the asymptotic property of stability. The initial state can also be random, as long as it is independent of the stochastic parameters, without affecting the analysis.

REFERENCES

1. W.L. DeKoning, Ph.D. Thesis, Delft University Press, 1984.
2. E. Yaz, "Stabilization of deterministic and stochastic-parameter discrete systems," Int. J. Control, Vol. 42, pp. 33-41, 1985.
3. E. Yaz, "Further results on the stabilizing property of certainty equivalent controllers," IEEE Trans. Autom. Contr., Vol. AC-31, pp. 586-587, 1986.

THE EFFECT OF LASER LINEWIDTH AND PHASE DISTORTION IN A CASCADED ILL REPEATER SYSTEM ON SYSTEM PERFORMANCE

VINCENT O'BYRNE
School of Electronic Engineering
University College of North Wales, U.K.

ABSTRACT

In this paper, the effect of the non-linear phase distortion associated with amplifying a FSK signal by means of an Injection-Locked Laser (ILL) together with the influence of the finite transmitter laser linewidth and the internally generated noise of the ILL on system performance are considered in a repeatered system. It is found that the aforementioned distortion and noise can cause an increase in the required signal to noise ratio (SNR) at the receiver.

I INTRODUCTION

The use of coherent systems offer potentially the benefit of between 10-20dB improvement in receiver sensitivity over direct detection systems[1]. However the benefit has proven difficult to obtain in practise owing to problems of finite laser linewidth, frequency and phase stability requirements of the local oscillator etc. A novel "hybrid" system incorporating the benefit of coherent transmission with the relative simplicity of direct detection by employing pre-detection optical discrimination is a possible optical fibre communication system, as shown in figure 1. This optical FM-AM conversion can be carried out by employing an interferometer type discriminator[2]. A system, so based would in theory produce shot noise limited operation since the noise at the receiver is largely dependent on the relatively high power level of the ILL employed as a pre-amplifier prior to optical discrimination. This benefit could be obtained at the expense of only minor increased system complexity.

Such a system could employ ILLs as non-regenerative intermediate repeaters at regular intervals along the fibre optic channel. However in a repeatered system employing ILLs as intermediate optical amplifiers phase distortion is produced[3]. In general the linewidth of the transmitter laser has also a finite value[4] and noise is also generated within the ILLs themselves[12]. The result of these three "parameters" may cause system performance to deteriorate drastically in a repeatered system. To the author's knowledge no previous publication to date has considered these sources of phase noise and distortion in conjunction when describing the performance of the ILL in the above system. It is the aim of this paper to address the effect of these possible sources of error on overall system performance.

II THE INJECTION LOCKED LASER

The behaviour of the ILL is similar in nature to that of the Injection -Locked Oscillator (ILO) as described by Adler[5]. However, the refractive index of the ILL's active region depends on the injected current density, a phenomenon not considered by the aforementioned model[5]. This dependency results in an asymmetric locking region and the occurrence of both stable and unstable locking regions[6] within the locking bandwidth. These phenomena have been observed by many authors[e.g. 7]. This dependency of the active region's refractive index on the current density and the occurrence of both stable and unstable locking regions can be accounted for by the use of appropriate rate equations. One such model has been shown to be that derived by Lang[6]. This model comprises of three rate equations representing the variation of the internal cavity's complex field E_o, the current and Photon densities.

The distortion for an FSK signal which is amplified by an ILL has been shown previously[3], based on the model as derived by Lang [6], for filters at the transmitter and receiver of impulse responses h_T (t) and h_R (t) respectively, to be given by the set of equations;

$$\theta'_{norm}(t) = \frac{-T}{2\pi\beta_{fm}} \left\{ \frac{fd}{\Delta f_L} \frac{dm(t)}{dt} \left[\frac{1}{\sqrt{1-\left(\frac{2\pi fdm(t)+\Delta\omega_o}{\Delta\omega_L}\right)^2}} - \frac{1}{\sqrt{1-\left(\frac{\Delta\omega_o}{\Delta\omega_L}\right)^2}} \right] \right\} * h_R(t) \tag{1}$$

$$m(t) = \sum_i b_i \mathrm{rect}\left(\frac{t-iT}{T}\right) * h_T(t) \tag{2}$$

$$\Delta\omega_L = 2\pi\Delta f_L = \frac{k\, E_{EXT}}{E_o} \sqrt{1+R^2} \tag{3}$$

NOTE: Quasi-stationary conditions are assumed to apply to the system under consideration.

where fd is the maximum signal frequency deviation, $\Delta\omega_o = 2\pi\Delta f_o$ is the relative detuning between the mean input angular frequency ω_c and the cavity resonant frequency of the ILL at threshold ω_{th}, m(t) is the message signal as applied to the transmitter laser, rect(t) $\triangleq$ 1, $|t| \leqslant 1/2$, zero elsewhere, $b \in \{+1,-1\}$, Δf_L is the half locking bandwidth. E_{EXT} and E_o are the injected field amplitude just after passing through the input facet of the ILL and the ILL field amplitude within the cavity, assumed constant; R is the linewidth enhancement factor. β_{fm} = fd/B_r, B_r is the bit rate and equals the reciprocal of the pulse interval T, k is the input coupling coefficient.

In equation 3 and the following it is assumed that the amplitudes E_{EXT} and E_o are all normalized so that their absolute values squared gives the corresponding photon densities.

III ANALYSIS OF THE PHASE DISTORTION

In the 'hybrid' system under consideration we must incorporate the receiver and transmitter filters in the analysis as they have been found to effect the level of the phase distortion[8]. In general their will be some parasitic filtering of the transmitted signal prior to launch due to circuit elements etc. As an illustrative example in our analysis of the phase distortion we shall assume that the transmitter filter can be approximated by an ideal lowpass filter with a frequency cut-off at six times the Nyquist rate. The receiver filter is usually taken to be such that the Fourier Transform of the signal, in the frequency domain, applied to the decision circuit is a 100% raised cosine.

To obtain the distortion at the sampling times we represent the transmitted signal by a test sequence of arbitrary length. This base sequence is cascaded with the inverse of its time reversed sequence and assumed periodic. This ensures that over all test sequences the mean instantaneous frequency deviation is zero. This is important since it was an assumption in the derivation of equation 1, that the mean frequency of the transmitted signal equals the carrier frequency of the unmodulated signal. We employ equations (1-3) and vary the period of the test sequence until we observe convergence in the maximum and standard deviation of the distortion at the sampling times. An illustrative example of the phase distortion is shown in figure 2 for the periodic sequence 10000011110.

As pointed out in Henry et al[13] the locking region is confined to;

$$-\frac{k\,E_{EXT}\sqrt{1+R^2}}{E_o} \leq d \leq \frac{k\,E_{EXT}}{E_o} \qquad (4)$$

where d represents nominal detuning from the threshold relaxation frequency ω_{th}.

Only part of the locking region is stable [6] and therefore normal operation is chosen to be confined to this region. Figure 3, based on illustrative parameters as in table 1 [10] for a 1.55µm InGaAsP Fabry-Perot laser amplifier with R=-6 and a fractional excess pumping r of 0.3, shows the variation of the stable, total locking bandwidth and the resultant shift in the central frequency of the stable region (Δf_o). It shall be assumed in the following analysis that the carrier frequency of the transmitted signal coincides with the centre frequency of the stable locking region as defined by Lang's model[6].

IV THE NOISE OUTPUT OF AN ILL

To investigate the noise at the receiver due to the linewidth of the transmitter laser and the distortion produced by the ILL we must first consider the noise spectrum of the transmitter laser and the effective intermediate transfer functions of the cascaded ILLs as seen by the phase noise. The FM noise $<|F_n(f)|^2>$ has been shown, by employing Langevin Force terms [12], to be given by;

$$<|F_n(f)|^2> = \frac{\beta}{2\tau_p r\pi}\left\{1+\frac{(\omega\Delta\omega_{Lo}\sin\theta_o)^2+(\omega\Delta\omega_{Lo}\sin\theta_o+R\omega_r^2)^2}{\omega^2(\Delta\omega_{Lo}\cos\theta_o+\omega_s)^2+(\omega_r^2+\Delta\omega_{Lo}\,\omega_s\cos\theta_o-\omega^2)^2}\right\} \qquad (5)$$

$$\omega_S = \frac{1}{\tau_S} + G_N E_o^2\,;\quad \omega_r^2 = G_N G(N)\bar{E}_o^2;\quad \Delta\omega_{Lo} = \frac{K\,E_{ext}}{E_o}$$

where ω_r is the resonant frequency, $\Delta\omega_{Lo}$ is the half locking bandwidth for $R = 0$, θ_o is the locked phase of the ILL for an unmodulated input signal; β represents the coefficient of spontaneous emission, τ_p and τ_s the photon and carrier lifetimes respectively. G_N is the rate of change of modal gain with carrier density, G(N) is the modal gain at the operating frequency of the laser; r the excess pumping rate.

The effective transfer function or noise suppression factor (S1) for input phase noise $(\overline{\Delta f_{IN}^2})$ to the ILL can be approximated for low frequencies relative to ω_r as [11,12];

$$S1 = \frac{1}{1 + \left(\frac{\omega}{\Delta\omega_L}\right)^2} \tag{6}$$

$$\Delta\omega_L = \frac{k\,E_{EXT}}{E_o}\sqrt{1+R^2}\;\mathrm{COS}\,(\theta_o - \tan^{-1}R) \quad .$$

For a filter at the receiver of impulse response $h_R\,(t)$ the variance of the noise accompanying the signal as applied to the decision circuit is given by;

$$\sigma^2 = \frac{1}{(2\pi fd)^2}\int_{-\infty}^{\infty} |H_R(f)|^2\,\{S1(\overline{\Delta f^2_{IN}}) + (1-S1)(\overline{\Delta f^2_a})\}\,df \tag{7}$$

where the noise has been normalised to 2 πfd times the maximum frequency deviation, no bandlimiting of the signal has been assumed prior to filtering at the receiver and $H_R(f)$ is the Fourier Transform of $h_R(t)$.

V NOISE IN A CASCADED SYSTEM

The noise from the rth ILL of a cascade system is given as [11];

$$\overline{(\Delta f^2_o)}_r = \overline{(\Delta f^2_a)}_r(1-S1) + \overline{(\Delta f^2_{IN})}_r\,S1 \tag{8}$$

where

$$\overline{(\Delta f^2_{IN})}_r = \overline{(\Delta f^2_o)}_{r-1}$$

and $\overline{(\Delta f^2_o)}\overline{(\Delta f^2_{IN})}\overline{(\Delta f^2_a)}$ refers to the output, the input and internal noise sources respectively. The subscripts refer to the particular ILL under consideration. For the case of N repeaters in cascade the total "channel" noise as seen by the detector is given by;

$$\sigma^2 = \frac{1}{(2\pi f_d)^2}\int_{-\infty}^{\infty} |H_R(f)|^2\{(1-S1^N)\,\overline{(\Delta f^2_a)} + S1^N\,\overline{(\Delta f^2_o)}_1\}\,df \tag{9}$$

where all ILLs are assumed similar in nature and operating under similar conditions.

VI SYSTEM PERFORMANCE

In the system under consideration where the receiver responds to the instantaneous frequency deviations of the input signal we may represent the distortion as given by equation (1) as a form of intersymbol interference (isi). This facilitates the employment of Glave's Bound [9] in order to obtain the required Signal to Noise ratio (SNR) at the receiver for a given required Probability of error (Pe). The noise due to the finite linewidth of the transmitter laser can be shown to be gaussian and therefore the total noise at the receiver can be given as the addition of the variances of the transmitted noise and receiver noise.

In our analysis we will assume an Illustrative parameter set for the ILL. These parameters are given in table 1 [10] where the linewidth enhancement factor and fractional excess pumping rate are as previously given. The external power gain of the ILL will be taken as 30dB, the value of β_{fm} will be assumed equal to unity.

We obtain by employing equations (1-3) that the maximum distortion and standard deviation at the sampling times are given as in figure 4 as a function of the maximum frequency deviation , for an ILL power gain of 30dB. By employing the Quasi-stationary hypothesis we shall assume that the distortion increases linearly with the number of repeaters.

Assuming parameters for the transmitter (master) laser as given previously together with a reduced linewidth enhancement factor R of -2 and biased at twice the threshold current, the FM spectrum of the unmodulated signal and ILL (slave) may be represented as shown in figure 5.

Figure 6 represents the normalized noise variance as seen by the decision circuit at the receiver due to the transmitted and "channel" noise. In figure 7 the variation of the Probability of error (Pe) against Signal to noise ratio (SNR) is shown for various values of the maximum frequency deviation (fd), where the number of repeaters has been taken as a parameter.

VII CONCLUDING REMARKS

It is clearly seen from figures (4-7) that the phase distortion and laser linewidth can cause a system penalty to be incurred over ideal transmission. From figures 7 (a)-(c) it is clearly seen that in employing this type of system careful consideration will have to be given to appropriate choice of maximum frequency deviation.

This paper can be considered as representing the system under the influence of phase noise and distortion alone. However it has been found that channel intensity noise can be quite significant. This noise could, together with the present subject matter, cause imitations to the usefulness of the ILL in repeatered systems. This area of research is at present being investigated.

ACKNOWLEDGEMENT

I would like to express my gratitude to the University College of North Wales, Bangor for a postgraduate Studentship, SERC for further support and Prof. J.J. O'Reilly for many useful discussions.

REFERENCES

1. Okoshi, T., K. Emura, K. Kikuchi and R. Th. Kersten, "Computation of Bit-error rate of various Heterodyne and Coherent Type Optical Communication Schemes, " J. of Optical Commun., vol. 2, No. 3, pp. 89-96, 1981.

2. Saito, S., Y. Yamamoto, and T. Kimura, "S/N and Error rate evaluation for an optical FSK-hererodyne Detection System using Semiconductor Lasers," IEEE J. Quantum Electron., vol. QE-19, pp. 180-193, Feb., 1983.

3. O'Byrne, V., " Phase Distortion in an Injection-Locked Laser," To be Published

4. Henry, C.H., "Theory of the Linewidth of Semiconductor Lasers," IEEE J. Quantum Electron., vol. QE-18, pp. 259-264, Feb., 1982.

5. Adler, R., "A study of Locking Phenomena in Oscillators," Proc. IRE, vol. 34, pp. 351-357, May-June, 1946.

6. Lang, R., "Injection-Locking properties of a Semiconductor Laser," IEEE J. Quantum Electron., vol. QE-18, No. 6, pp. 976- 983, June, 1982.

7. Goldberg, L., H.F. Taylor, and J.F. Weller, "Locking Bandwidth Asymmetry in Injection-locked GaAlAs Lasers," Electron. Lett., vol. 18, No. 23, pp.986-987, Nov., 1982.

8. O'Byrne, V., "Long Haul Fibre Optic Communications Exploiting the advantages of Coherent and Non-coherent Transmission," MSc. dissertation, Nov. 1985. University of Essex

9. Glave, F., "An Upper Bound on the Probability of Error due to Intersymbol Interference for correlated Digital Signals,"IEEE Trans. on Info. Theory, vol. IT-18, No. 3, May, 1972.

10. Brown, G.N., "A study of the Static locking Properties of Injection Locked Laser Amplifier," Br. Telecom Technol. J., vol. 4, No.1, pp. 71-80, Jan. 1986.

11. Hines, M.E., J-C. R. Collinet, and J.G. Ondria, "FM noise suppression of an Injection Phase-locked Oscillator, " IEEE Trans. on Microwave Theory Tech., vol. MTT-16, No. 9, pp. 738-742, Sept. 1968.

12. Saito, S., F. Mogensen, and H. Olesen, "Effective bandwidth for FM noise suppression in an Injection-locked Semiconductor Laser," Electron. Lett, vol. 21, No. 24, pp. 1173-1175, Nov. 1985.

13. Henry, C.H., N.A. Olsson, and K. Duttan, "Locking range and stability of Injection-Locked 1.54 m InGaAsP Semiconductor Lasers," IEEE J. Quantum Electron., vol. QE-21, No. 8, pp. 1152-1157, Aug., 1985.

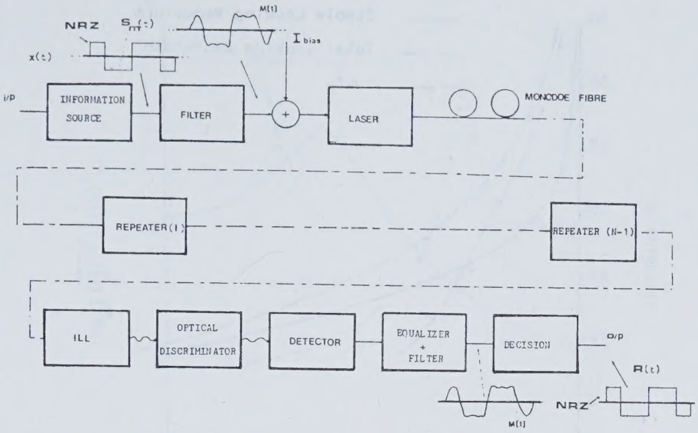

NRZ
S(t)
M(t)
I bias
x(t)
i/p
INFORMATION SOURCE
FILTER
+
LASER
MONODOE FIBRE
REPEATER (I)
REPEATER (N-1)
ILL
OPTICAL DISCRIMINATOR
DETECTOR
EQUALIZER + FILTER
DECISION
o/p
R(t)
NRZ
M(t)

Figure 1 Diagram of the "hybrid" system under consideration which employs optical discrimination.

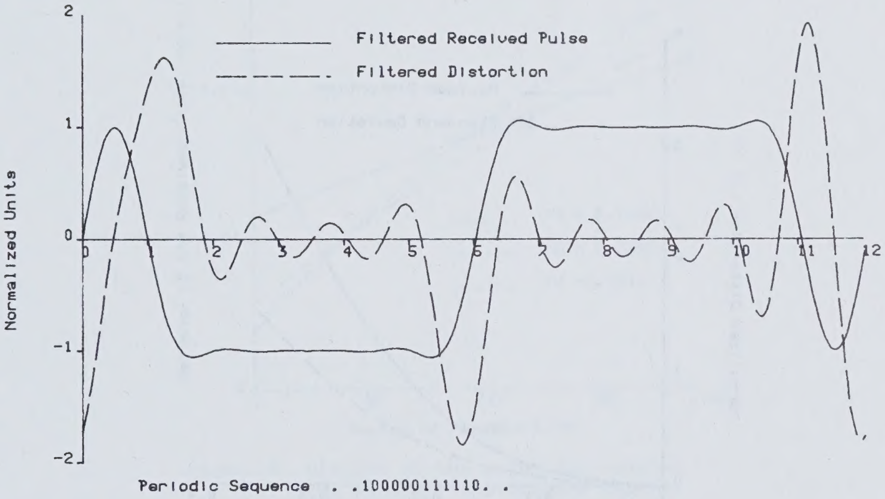


Figure 2 Illustrative diagram of the distortion (unit=10^{-4}), and filtered pulse (unit=0.1GHz) as applied to the decision circuit.

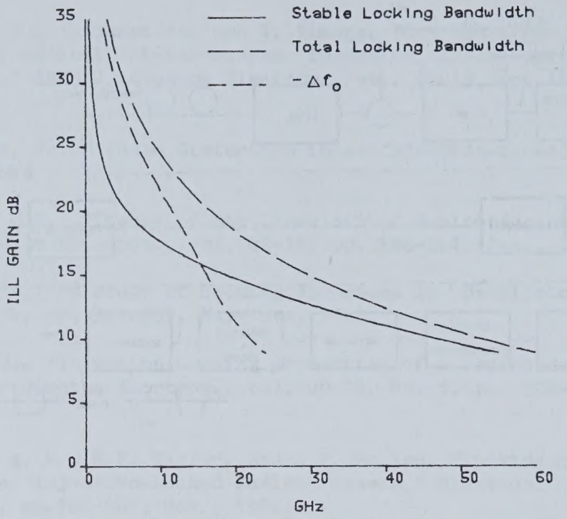


Figure 3 Diagram of the stable locking bandwidth, the frequency shift of the central frequency of the stable locking region and the total locking region against injection locked laser power gain.

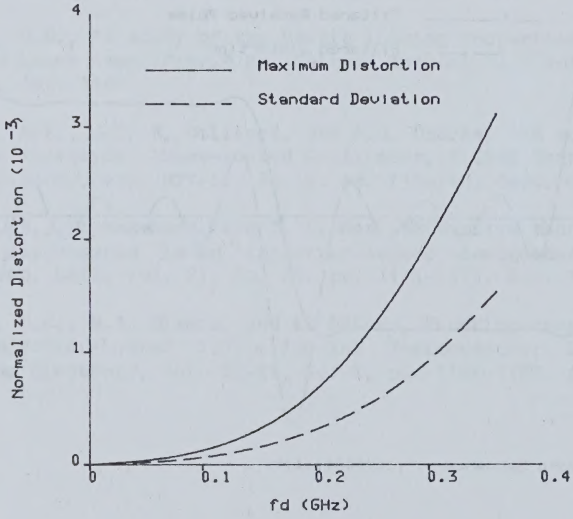


Figure 4 Diagram of the variation of the distortion with the maximum signal deviation (fd).

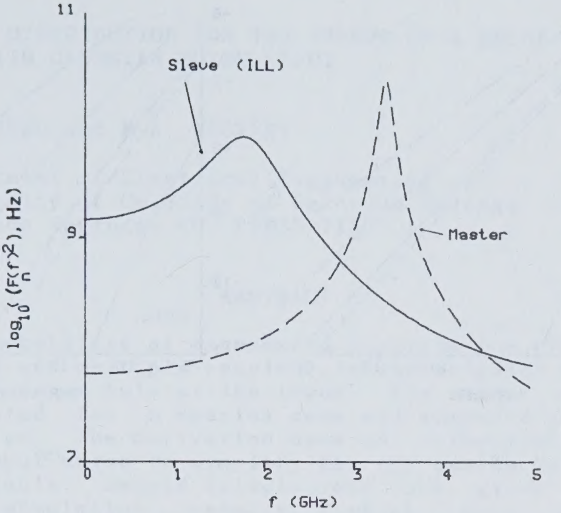


Figure 5 Diagram of the transmitter (Master) and repeater (Slave) laser FM noise spectrum.

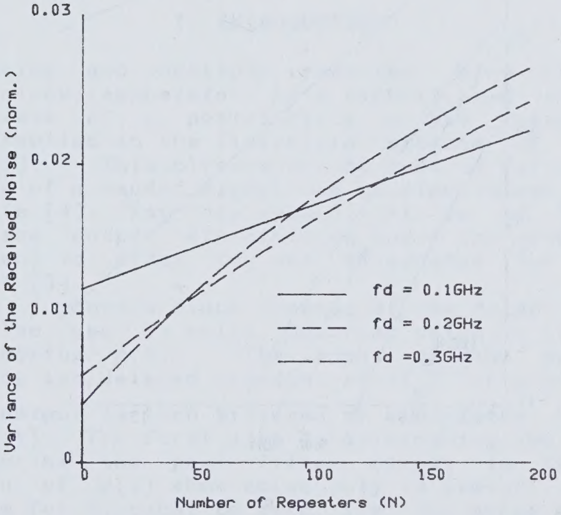


Figure 6 Diagram of the noise variance at the receiver against the number of intervening repeaters, as a function of the maximum signal deviation fd.

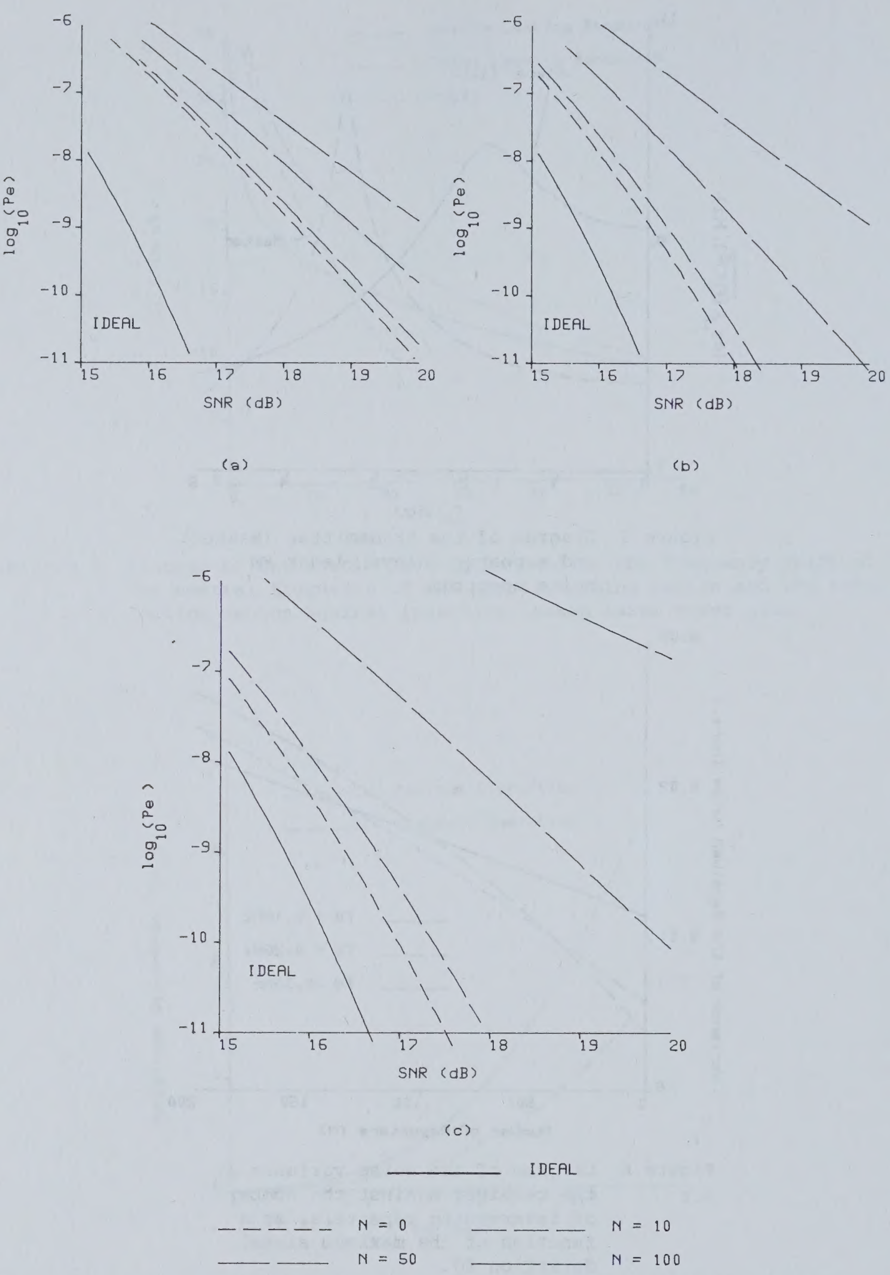


Figure 7 Probability of error (Pe) against required Signal to Noise Ratio (SNR) for the three cases (a) fd=0.1GHz, (b) fd=0.2GHz and (c) fd=0.3GHz as a function of N, the number of cascaded repeaters.

EXCEEDANCE DISTRIBUTION FOR THE OUTPUT OF A DELAY AND MULTIPLY RECEIVER WITH GAUSSIAN NOISE INPUT

D.E. REED and M.A. WICKERT

Department of Electrical Engineering
University of Colorado at Colorado Springs
Colorado Springs, CO 80933-7150

ABSTRACT

The probability of exceedance distribution of the output of a delay and multiply receiver is investigated for the case of Gaussian noise only at the input. The exact distribution is calculated for a special case and compared to a Gaussian approximation. The derivation uses an orthonormal expansion of the input noise to simplify the expression for the output random variable. Sample calculations are given along with computer simulation results, which show the Gaussian approximation to be good for exceedance probabilities > 10^{-5} with an FIR post filter of length N=64.

I. INTRODUCTION

The delay and multiply receiver, also known as the delay-line clock regenerator, is a circuit used to determine the baud rate of a periodically pulsed signal. It has received attention in the literature because of its simple design [1-3]. This circuit can be used to initially detect the presence of a bauded signal and to simultaneously estimate the baud rate [4]. For this reason, it is of interest to determine the output distribution under the condition of no signal present in order to set thresholds for a Neyman-Pearson test [5].

Figure 1 shows a block diagram of the delay and multiply receiver. The input is white, Gaussian noise, x(t), which is filtered, giving n(t). The ouput of the prefilter is multiplied by its delayed version, $n(t-T_d)$, to give z(t). The multiplier output is then filtered by a bandpass post filter to give y(t). The first step in determining the performance of a detector at the post filter output is to find the distribution of y(t) when noise only is present. Kac solved this problem for T_d equal to zero; i.e. the delay and multiply is replaced with a squaring operation [6]. Work on delay and multiply statistics has concentrated on the second order moments [7]. When T_d is not zero, the analysis is complicated by the inclusion of the effects of correlation between n(t) and $n(t-T_d)$ as well as the phase shift introduced at the bandpass center frequency by the delay T_d.

The problem is addressed here in two cases. First, when the phase shift of $\omega_o T_d$, ω_o being the center frequency of the receiver, is a multiple of 2π, a closed form expression is given for the output pdf. Second, when this is not the case, moments are given which allow the output distribution to be approximated by a Gaussian Quadrature Rule (GQR) [8-9]. Here sample results are calculated only for the first case. These results are compared to the use of a Gaussian approximation for y(t). Also, histograms obtained through computer simulation are presented and compared with the calculated distributions.

Section II gives the mathematical models of the circuit and its signals, and the distribution of y(t) is derived for a sampled signal system. In Section III, results of calculations for the output probability distributions are given and compared to the Gaussian approximation and to simulation results for typical system parameters. Finally, Section IV gives conclusions and suggests approaches for completing the problem of computing detection performance.

II. SYSTEM MODEL

Refering to Figure 1, we define the input noise signal to be

$$x(t) = x_c(t)\cos(\omega_0 t) + x_s(t)\sin(\omega_0 t) \tag{1}$$

The prefiltered signal n(t) is given by

$$n(t) = n_c(t)\cos(\omega_0 t) + n_s(t)\sin(\omega_0 t) \tag{2}$$

where n_c and n_s are independent, lowpass, stationary Gaussian processes. The scaled output of the multiplier is

$$\begin{aligned} z(t) &= n_c(t)n_c(t-T_d)\cos\theta + n_c(t)n_s(t-T_d)\sin\theta \\ &\quad -n_s(t)n_c(t-T_d)\sin\theta + n_s(t)n_s(t-T_d)\cos\theta \end{aligned} \tag{3}$$

where $\theta = \omega_0 T_d$. To simplify the calculations we assume that z(t) is a sampled signal. Defining the filters to have finite impulse responses we have;

$$n_c(t) = \sum_{i=1}^{N} x_c(i)\ Q(t-i) \tag{4}$$

$$n_s(t) = \sum_{i=1}^{N} x_s(i)\ Q(t-i) \tag{5}$$

and

$$y(t) = \sum_{i=1}^{N} z(i)\ K(t-i) \tag{6}$$

The pdf of z(t), for $\rho \neq \pm 1$, can be found to be [10]

$$f_z(z) = \begin{cases} \frac{1}{2\sigma^2}\, \ell^{-\frac{z}{\sigma^2(1+\rho)}} & z > 0 \\ \frac{1}{2\sigma^2}\, \ell^{\frac{z}{\sigma^2(1-\rho)}} & z < 0 \end{cases} \tag{7}$$

where σ^2 is the variance of $n_c(t)$ and ρ is given by

$$\rho = E\{ n_c(t)\, n_c(t-T_d) \} / \sigma^2 . \tag{8}$$

For $\theta \neq 0$ one can show the moments of $z(t)$ to be given by the following;

$$E\{z^k\} = \sum_{i=0}^{k} \sum_{j=0}^{i} \sum_{n=0}^{k-i} \binom{k}{i}\binom{i}{j}\binom{k-i}{n}(-1)^n(\cos\theta)^{k+j-i-n} \cdot (\sin\theta)^{i+n-j}\, m_{i,j+n}\, m_{k-i,k-n-j} \tag{9}$$

where $m_{i,j}$ can be found by

$$m_{i,j} = \sum_{n=0}^{i} \rho^{i-n}\, m_n[\sigma^2(1-\rho^2)]\, m_{i+j-n}[\sigma^2] \tag{10}$$

with

$$m_k[\sigma^2] = \begin{cases} \sigma^k \frac{k!}{2^{k/2}(\frac{k}{2})!} & k \text{ even} \\ 0 & k \text{ odd} \end{cases} \tag{11}$$

With these moments the distribution of $z(t)$ can be approximated by using the GQR [8].

In 1947, Kac found the pdf of a filtered, squared IF signal [6]. His approach was to expand the input signal onto an orthonormal set of basis functions. This simplified the expression for the output to a sum of independent r.v.s;

$$y = \sum \lambda_k z_k \quad , z_k \text{ are i.i.d.}$$

where λ_k are eigenvalues of an integral equation. In this study the system is the same except that the squaring operation is replaced by the delay and multiply operation, and the signals are discrete time. The problem is broken into two cases:

Case I: $\theta = 0$.

For the case of $\theta=0$ we have the expression for z as

$$z(t) = n_c(t)n_c(t-T_d) + n_s(t)n_s(t-T_d) \tag{12}$$

If this equation is written in terms of x_c and x_s, the quardature components of the input "white" noise, we have;

$$z(t)=\sum_{i=1}^{N} \sum_{j=1}^{N} Q(t-i)Q(t-j)[x_c(i)x_c(j-T_d) + x_s(i)x_s(j-T_d)] \tag{13}$$

where Q(t) is the prefilter unit pulse response. The post filter output is ;

$$y(t) = \sum_{i=1}^{N} K(t-i)\ z(i) \tag{14}$$

Now expand x_c and x_s onto an orthonormal basis set for the observation interval and obtain;

$$x_c(t) = \sum_{i=1}^{N} u_i\ \phi_i(t) \qquad\qquad x_s(t) = \sum_{i=1}^{N} v_i \cdot \phi_i(t)$$

$$x_c(t-T_d) = \sum_{i=1}^{N} u_i'\ \phi_i(t) \qquad\qquad x_s(t-T_d) = \sum_{i=1}^{N} v_i'\ \phi_i(t) \tag{15}$$

Since $\{\phi_i(t)\}$ is an orthonormal basis, then u_i and v_i are independent Gaussian random variables. Then for some arbitrary time t we have

$$y = \sum_{i=1}^{N} \sum_{j=1}^{N} (u_i u_j' + v_i v_j') \sum_{m=1}^{N} \sum_{n=1}^{N} \Lambda(m,n)\ \phi_i(m)\ \phi_j(n) \tag{16}$$

where

$$\Lambda(m,n) = \sum_{i=1}^{N} K(i)\ Q(m-i)\ Q(n-i) \tag{17}$$

is the matrix with eigenvales and eigenvectors defined by

$$\sum_{n=1}^{N} \Lambda(m,n)\ \phi_i(n) = \lambda_i\ \phi_i(m) \tag{18}$$

Thus,

$$y = \sum_{i=1}^{N} (u_i u_i' + v_i v_i')\ \lambda_i = \sum_{i=1}^{N} z_i\ \lambda_i \tag{19}$$

where the z_i are independent r.v.s with pdfs given in (7).

Unfortunately, the correlation coeficients ρ_i, between u_i, v_i and u_i', v_i', are different, in general, for each i. They are given by;

$$\rho_i = \frac{E\{u_i u_i'\}}{\sigma^2} = \sum_{n=1}^{N} \phi_i(n)\ \phi_i(n+T_d) \tag{20}$$

Thus, ρ_i must be calculated for all i in order to obtain the pdf of y. By transforming (7) and performing a partial fraction expansion we find the characteristic function of y to be

$$\Phi_y(\omega) = \sum_{k=1}^{N} \left[\frac{A_k^1}{\frac{1}{1-\rho_k} + j\lambda_k \omega} + \frac{A_k^2}{\frac{1}{1+\rho_k} - j\lambda_k \omega} \right] \tag{21}$$

in which

$$A_k^1 = \frac{1}{2} \prod_{\substack{l=1 \\ l \neq k}}^{N} \frac{1}{\left(1 - \frac{\lambda_l(1-\rho_l)}{\lambda_k(1-\rho_k)}\right)\left(1 + \frac{\lambda_l(1+\rho_l)}{\lambda_k(1-\rho_k)}\right)} \tag{22}$$

$$A_k^2 = \frac{1}{2} \prod_{\substack{l=1 \\ l \neq k}}^{N} \frac{1}{\left(1 - \frac{\lambda_l(1+\rho_l)}{\lambda_k(1+\rho_k)}\right)\left(1 + \frac{\lambda_l(1-\rho_l)}{\lambda_k(1+\rho_k)}\right)} \tag{23}$$

Therefore, the pdf of y is

$$f_y(y) = \begin{cases} \displaystyle\sum_{\lambda_k > 0} \frac{A_k^1}{\lambda_k} e^{\frac{-y}{\lambda_k(1-\rho_k)}} + \sum_{\lambda_k < 0} \frac{-A_k^2}{\lambda_k} e^{\frac{y}{\lambda_k(1+\rho_k)}} & y > 0 \\ \displaystyle\sum_{\lambda_k < 0} \frac{-A_k^1}{\lambda_k} e^{\frac{-y}{\lambda_k(1-\rho_k)}} + \sum_{\lambda_k > 0} \frac{A_k^2}{\lambda_k} e^{\frac{y}{\lambda_k(1+\rho_k)}} & y < 0 \end{cases} \tag{24}$$

Case II: $\theta \neq 0$

In the more general case of $\theta \neq 0$ we again have the expression;

$$\begin{aligned} z(t) = {} & n_c(t)n_c(t-T_d)\cos\theta + n_c(t)n_s(t-T_d)\sin\theta \\ & - n_s(t)n_c(t-T_d)\sin\theta + n_s(t)n_s(t-T_d)\cos\theta \end{aligned} \tag{25}$$

In terms of x_c and x_s it is

$$z(t) = \sum_{i}\sum_{j} Q(t-i)\, Q(t-j)\, [\, x_c(i)x_c(j-T_d)\cos\theta + x_s(i)x_c(j-T_d)\sin\theta - x_c(i)x_s(j-T_d)\sin\theta + x_s(i)x_s(j)\cos\theta\,] \tag{26}$$

Again we expand x_c and x_s onto the basis set $\{\phi_i(t)\}$ and write

$$y = \sum_{i=1}^{N} \sum_{j=1}^{N} (u_i u'_j \cos\theta + u_i v'_j \sin\theta - v_i u'_j \sin\theta + v_i v'_j \cos\theta) \cdot \sum\sum \Lambda(m,n)\, \phi_i(m)\, \phi_j(n) \tag{27}$$

Thus

$$y = \sum_{i=1}^{N} (u_i u'_i \cos\theta + u_i v'_i \sin\theta - v_i u'_i \sin\theta + v_i v'_i \cos\theta)\, \lambda_i$$

$$= \sum_{i=1}^{N} z_i\, \lambda_i \tag{28}$$

where the z_i are independent r.v.s with moments found by(9). Unfortunately, the characteristic functions of these z_i are not known. However, if the moments for each z_i are found by (9), then one can find the moments for y. This is done as

$$y_n = \sum_{i=1}^{n} \lambda_i\, z_i \tag{29}$$

$$E\{\, y_n^k \,\} = \sum_{i=0}^{k} \binom{k}{i} \lambda_n^i\, E\{\, y_{n-1}^{k-i} \,\}\, E\{\, z_n^i \,\} \tag{30}$$

where $E\{\, z_n^i \,\}$ is a function of ρ_i.

III. RESULTS

The equations of Section II, Case I, can now be used to calculate the output exceedance distribution. It is of interest to compare the actual distribution to a Gaussian approximation. Results from a computer simulation of the delay and multiply receiver are also given here. Typical system parameters are used in this comparison. The prefilter impulse response Q(t) is taken to be exponential, $Q(t) = \ell^{-t/\tau}u(t)$, and the post filter is taken as $K(t) = \cos(2\pi Ft)u(t)u(N-t)$.

Only results for $\theta=0$ are given here. Figure 2 shows distributions for τ equal to 1.6, T_d equal to one sample, F= N/4 and values of N = 16, 32 and 64. For values of exceedance probability below 10^{-5} and N=16, the Gaussian assumption is obviously not good. For N larger than 64, the Gaussian approximation may be a reasonable one. Figure 3 shows the same curves with the histograms obtained from the simulation added. The simulation results show reasonable agreement with the theoretical results.

IV CONCLUSIONS

Several conclusions can be drawn from this analysis. For post filters with short impulse responses, the Gaussian approximation can be quite bad. As expected, the Gaussian approximation becomes better with increasing length, N, of the impulse response. The results of the calculations indicate that the Gaussian approximation for exceedance probabilities on the order of 10^{-5} is good for N > 64. Further extensions of the present work include; finding or bounding the exceedance probabilities for an envelope detector at the post filter output, and finding the exceedance probabilities for the case of the signal being present.

References

1 A.N. D'Andrea and U. Mengali,"Performance Analysis of the Delay-Line Clock Regenerator," IEEE Trans. Commun., Vol. COM-34,pp.321-328, April 1986.

2 R.E. Ziemer and R.L. Peterson, Digital Communications and Spead Spectrum Systems, Macmillan Publishing Co., New York, 1985.

3 R. McCallister, "Generalized Cross-Spectrum Symbol Synchronization," Ph.D. Dissertation, Arizona State University, December 1981.

4 D.E. Reed and M.A. Wickert,"Minimization of Baud Rate Spectral Lines by Delay and Multiply Receivers," MILCOM '86, October 1986.

5 H.L. Van Trees, Detection, Estimation, and Modulation Theory Part 1, John Wiley & Sons, Inc., New York, 1968.

6 Mark Kac and A.J.F. Siegert,"On the Theory of Noise in Radio Receivers with Square Law Detectors,"J. of Appl. Phys.,Vol. 18 (April 1947),pp.383-97.

7 W.R. Bennet, "Statistics of Regenerative Digital Transmission," B.S.T.J., vol. 37, pp.1501-42,Nov. 1958.

8 M.H. Meyers,"Computing the Distribution of a Random Variable via Gaussian Quadrature Rules,"B.S.T.J.,Vol. 61, No. 9 (Nov. 1982),pp.2245-61.

9 G.H. Golub and J.H. Welsch,"Calculation of Gauss Quadrature Rules," Math. Comp.,26, No. 106 (April 1969), pp.221-30.

10 A. Papoulis,Probability, Random Variables, and Stochastic Processes,New York; McGraw-Hill, 1965.

Figure 1 Delay and Multiply Block Diagram

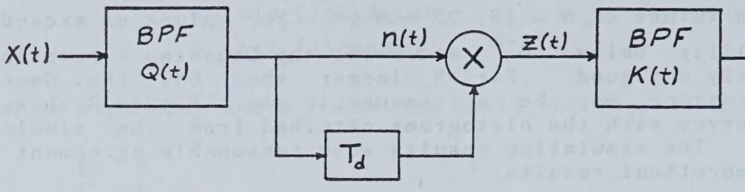

X(t)
BPF
Q(t)
n(t)
X
z(t)
BPF
K(t)
y(t)
T_d

Figure 2 Output exceedance; ρ=0.5, N= 16, 32 and 64.

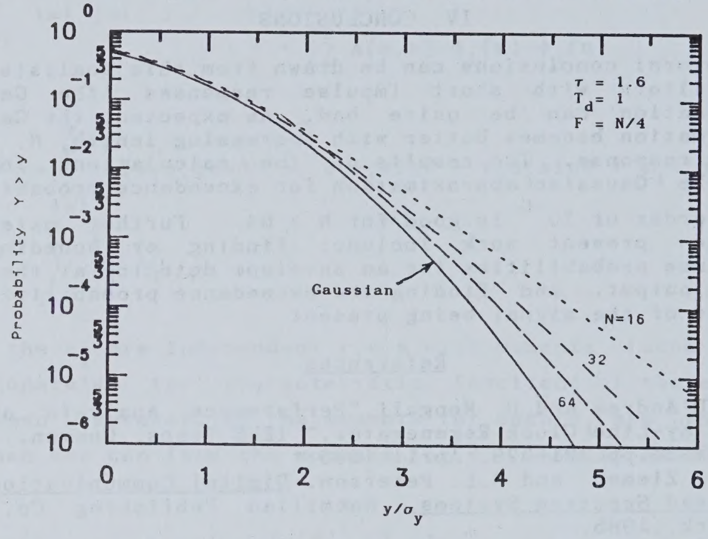
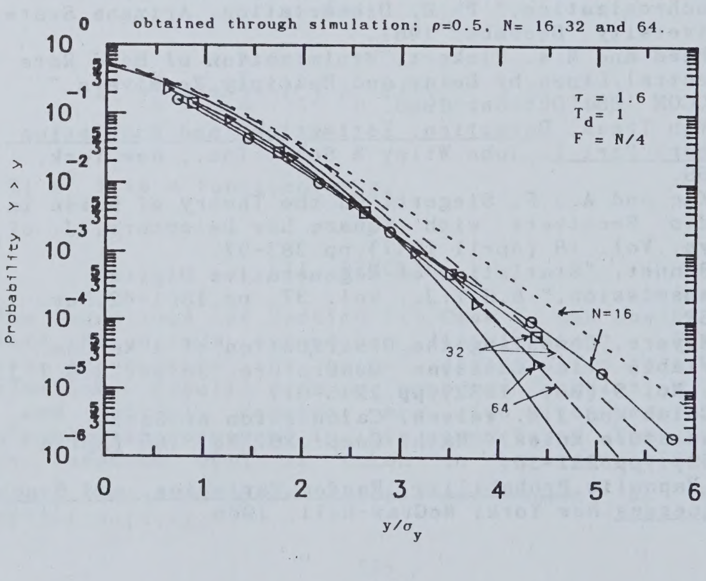


Figure 3 Exact exceedance probability versus statistics obtained through simulation; ρ=0.5, N= 16,32 and 64.



RATIO-STATISTIC COMBINING FOR CHANNELS WITH PARTIAL-BAND INTERFERENCE

CATHERINE M. KELLER
Dept. of Electrical and Computer Eng.
DSP Laboratory: Communications Sector
Northeastern University
Boston, MA 02115

ABSTRACT

Ratio-statistic combining is proposed for use in diversity transmission systems with partial-band interference. We find the exact probability of error for a receiver employing ratio-statistic combining, and compare it to the performance of a receiver with perfect side information and of a receiver which uses clipped-linear combining. We cover systems with noncoherent demodulation and binary orthogonal signaling. The partial-band interference is Gaussian, and we include Gaussian quiescent noise in the analysis to account for wideband noise sources. Some of the results apply to general (non-Gaussian) interference. We also give numerical results for a system with ratio-statistic combining in the presence of Rayleigh fading and partial-band interference.

INTRODUCTION

Various results have been published recently on the performance of diversity combining techniques for communications in the presence of partial-band interference. In [1-3], the performance is given for a diversity transmission system that employs side information regarding the presence or absence of the interference on the diversity receptions. The results in these papers demonstrate that coding and diversity are effective against partial-band interference as long as side information is available. The results in [4] show that the performance of systems with side information is fairly insensitive to the type of diversity combining scheme employed—the major performance gains in the diversity transmission system are due to the availability of the side information.

Side information is not always available at the receiver, and diversity combining techniques that do not require side information are of interest. The performance of systems without side information depends heavily on the type of diversity combining technique employed. This idea is demonstrated in [4-9]; in these papers, several diversity combining schemes that do not use side information are proposed, analyzed, and compared. In [4-6], it is shown that clipped-linear combining and the ratio-threshold test with majority logic decoding perform particularly well for channels with partial-band interference.

In clipped-linear diversity combining, the envelope detector outputs for each diversity reception are clipped before they are combined. The role of the clipper is to constrain the effects of strong narrowband interference. Although clipped-linear combining is very effective against partial-band interference, there is a practical disadvantage to this diversity combining scheme. The clipping level depends on the signal output voltage (i. e., the envelope detector output voltage due to signal only). This signal output voltage may be difficult to measure in practice.

A class of diversity combining schemes that avoids this practical disadvantage of clipped-linear combining uses ratio statistics. These ratio statistics are functions which involve ratios of the envelope detector outputs of the receiver. Two techniques studied in [5-6] use ratio statistics in a ratio-threshold test in order to determine whether a given diversity reception should be included in the decision statistics of the receiver. The use of ratio statistics in such a test was first proposed in [10], and an information theoretic study of a diversity combining scheme employing the ratio-threshold test is given in [11]. The ratio-threshold test with majority logic decoding is shown to be effective against partial-band interference in [5-6, 10-11].

The diversity combining technique proposed in this paper, namely, *ratio-statistic combining*, has several characteristics that either or both the ratio-threshold test with majority logic decoding and clipped-linear combining possess. Thus, it is of value to analyze the performance of ratio-statistic combining. In ratio-statistic combining, a clipping operation is performed in such a way that no clipping level has to be set. Thus, this diversity combining technique shares a useful property with clipped-linear combining, without the disadvantage. Ratio statistics are formed for each envelope detector output for each diversity reception such that the maximum value that any ratio statistic may have is 1. The receiver decision statistics are the sums of the ratio statistics over all of the diversity receptions.

In this paper, the exact probability of error is found for a system employing ratio-statistic combining. We examine how well the diversity combining scheme performs as a function of the partial-band interference duty factor ρ. In addition to the analysis of the system in Gaussian partial-band interference, we include a discussion of the performance of ratio-statistic combining in the presence of general, non-Gaussian interference. A comparison is made between ratio-statistic combining and clipped-linear combining. We use the optimum combining technique for a receiver with perfect side information to determine a lower bound on the performance.

SYSTEM MODEL

We consider a noncoherent communication system with binary orthogonal signaling and diversity level L. We assume that the diversity receptions are mutually independent. An example of such a system is a noncoherent frequency-hop spread-spectrum system with binary orthogonal signaling in which the diversity transmissions are sent over different frequency slots. The independence assumption is satisfied if the smallest spacing between frequency slots used by the diversity receptions of the same bit is larger than the coherence bandwidth of the communication channel. The partial-band interference is additive Gaussian noise and is present simultaneously at both of the filters of the receiver during a given diversity reception with probability ρ. With probability $1 - \rho$, the interference is not present on a diversity reception. In a frequency-hop system, ρ represents the fractional bandwidth occupied by the partial-band interference. The power spectral density due to the partial-band interference is $\frac{1}{2}\rho^{-1}N_I$ across that fraction of the band. Thus, the average power spectral density of the partial-band interference is $\frac{1}{2}N_I$.

We include additive quiescent noise in the analysis to account for wideband sources of interference. This quiescent noise, which is assumed to be Gaussian, has uniform power spectral density $\frac{1}{2}N_0$. Thus, on the fraction of the frequency band with interference, the total noise power spectral density is $\frac{1}{2}\rho^{-1}N_I + \frac{1}{2}N_0$. Let E_b denote the energy per bit, so that E_b/L is the energy per diversity reception. Then, we define the signal-to-noise ratio for a diversity reception with interference present to be

$$\begin{aligned} v_I \triangleq \frac{\beta}{\sigma_I} &= \sqrt{2(E_b/(\rho^{-1}N_I + N_0))/L} \\ &= \sqrt{2[\,[(\rho E_b/N_I)^{-1} + (E_b/N_0)^{-1}]\ L]^{-1}}. \end{aligned} \tag{1}$$

The parameter β is the envelope detector output voltage due to signal only, and the parameter $2\sigma_I^2$ is the variance of the envelope detector outputs due to interference and quiescent noise. On the fraction of the frequency band with partial-band interference absent, the total noise power spectral density is $\frac{1}{2}N_0$. The signal-to-noise ratio for a diversity reception with interference absent is

$$v_N \triangleq \frac{\beta}{\sigma_N} = \sqrt{2(E_b/N_0)/L}, \tag{2}$$

where $2\sigma_N^2$ is the variance of the envelope detector outputs due to the quiescent noise.

Let the receiver envelope detector outputs be denoted by $\{R_{k,l} : k = 0,1; 1 \leq l \leq L\}$. If the bit 0 is sent, $R_{0,l}$ is a Rician distributed random variable and $R_{1,l}$ is a Rayleigh distributed random variable. The receiver forms the ratio statistics $T_{0,l}$ and $T_{1,l}$ for each diversity reception:

$$T_{k,l} = \frac{R_{k,l}}{\max(R_{0,l}, R_{1,l})}, \quad k = 0,1. \tag{3}$$

Then, the ratio statistics are summed over l to form the receiver decision statistics:

$$\mathcal{W}_k = \sum_{l=1}^{L} T_{k,l}, \quad k = 0,1. \tag{4}$$

The bit corresponding to the larger decision statistic is chosen by the receiver. A block diagram for a receiver implementing ratio-statistic combining is shown in Figure 1.

Notice that the maximum value that $T_{k,l}$ can have is 1. For example, $T_{0,l}$ is equal to 1 whenever $R_{0,l} \geq R_{1,l}$, and it is equal to $R_{0,l}/R_{1,l} < 1$ otherwise. Thus, the statistics formed for each diversity reception can be thought of as clipped envelope detector outputs; the clipping is achieved by normalizing the envelope detector outputs with respect to the maximum envelope detector output. One consequence of this normalization is that interference present on a given diversity reception has a limited effect on the overall decision statistics.

SYSTEM ANALYSIS

Arbitrary Interference

The parameter ρ^* is defined as the largest number such that a given error probability can be achieved by the diversity transmission system in the presence of arbitrary interference with any duty factor less than this number [1]. This is a measure of the narrowband interference rejection capability of the system. It is defined so that it does not depend on the statistical distribution or the power level of the interference. The effects of background noise are excluded from the calculation of ρ^*.

First consider a receiver with perfect side information. In such a system, the receiver can ignore the diversity receptions that have interference and extract the data from the noise-free diversity receptions. If all of the diversity receptions have interference, then all of the diversity receptions are combined. The value of ρ^* for systems with perfect side information is $p_b^{1/L}$ [1]. That is, an error cannot occur unless interference is present on all diversity receptions.

In a receiver with clipped-linear combining, each envelope detector output is clipped at a level which is c times the signal output voltage, where c is a relative clipping level. The value of ρ^* for a system with clipped-linear combining depends on c as well as on the bit error probability and diversity level. The analysis of ρ^* for clipped-linear combining is given in [4]. If we consider $c = 1$, then ρ^* is the solution for ρ in

$$p_b = \sum_{j=\lfloor \frac{L+1}{2} \rfloor}^{L} \binom{L}{j} \rho^j (1-\rho)^{L-j}. \tag{5}$$

Interference must be present on half or more of the diversity receptions for an error to occur.

For a receiver with ratio-statistic combining, the worst situation for a diversity reception with interference is for $T_{0,l} = 0$ and $T_{1,l} = 1$ given that bit 0 is sent. Diversity receptions with interference absent have $T_{0,l} = 1$ and $T_{1,l} = 0$. Therefore, interference must be present on half or more of the diversity receptions for an error to occur. Thus, ρ^* for ratio-statistic combining is the solution for ρ in (5). The parameter ρ^* is always the same for clipped-linear combining with relative clipping level 1 and ratio-statistic combining.

Suppose that we require a bit error probability $p_b = 0.05$. If we have diversity level 1, the value of ρ^* for any of the receivers discussed is 0.05. If we have diversity level 3, the value of ρ^* for the receiver with perfect side information is 0.368, and it is 0.135 for a receiver with ratio-statistic combining or clipped-linear combining with $c = 1$. A system with $L = 7$ results in $\rho^* = 0.652$ for the receiver with perfect side information, and $\rho^* = 0.225$ for a receiver with ratio-statistic combining or clipped-linear combining. Note that ρ^* increases with each L for the receiver with perfect side information. For ratio-statistic combining, a value of L equal to $2N$, where N is an integer, gives no improvement over $L = 2N - 1$. This is because to calculate ρ^*, we assume that a tie always results in an error.

Gaussian Partial-Band Interference

The bit error probability for the system with ratio-statistic combining in the presence of partial-band interference is

$$p_b = \sum_{j=0}^{L} \binom{L}{j} \rho^j (1-\rho)^{L-j} P(e;j), \tag{6}$$

where $P(e;j)$ denotes the bit error probability given that j diversity receptions have interference present in addition to the quiescent noise. Let $W = \mathcal{W}_0 - \mathcal{W}_1$, where $\mathcal{W}_k$ is given in (4). If bit 0 is sent, an error is made if $W \leq 0$. Thus, we can find the average probability of error by evaluating the density function for W and then integrating the density function over $(-\infty, 0]$. The density function for the random variable W depends on how many diversity receptions have interference. Thus, to calculate $P(e;j)$, we consider the conditional density for W given that j diversity receptions have interference present.

Let $\{R_{k,l}^{I} : k = 1,2; l = 1,\ldots,j\}$ denote the envelope detector outputs given that interference is present. Similarly, $\{R_{k,l}^{N} : k = 1,2; l = j+1,\ldots L\}$ are the envelope detector outputs given that interference is absent. Also, let $\{T_{k,l}^{I} : k = 1,2; l = 1,\ldots,j\}$ and $\{T_{k,l}^{N} : k = 1,2; l = j+1,\ldots,L\}$ denote the ratio statistics given that interference is present and interference is absent, respectively. First, consider those diversity receptions with interference present, and let $X_l^I = R_{0,l}^I / R_{1,l}^I$. In terms of X_l^I, we can write the ratio statistics as

$$T_{0,l}^{I} = \min(X_l^I, 1)$$

$$T_{1,l}^{I} = \frac{1}{\max(X_l^I, 1)}. \tag{7}$$

Note that $T_{0,l}^I$ and $T_{1,l}^I$ are dependent random variables; if $X_l^I \geq 1$, then $T_{0,l}^I = 1$ and $T_{1,l}^I = 1/X_l^I$. On the other hand, if $X_l^I < 1$, then $T_{0,l}^I = X_l^I$ and $T_{1,l}^I = 1$.

Let $Y_l^I = T_{0,l}^I - T_{1,l}^I$. Then, we have

$$Y_l^I = \begin{cases} X_l^I - 1, & X_l^I < 1 \\ 1 - \dfrac{1}{X_l^I}, & X_l^I \geq 1. \end{cases} \tag{8}$$

We see from (8) that Y_l^I can have values in $[-1, 1]$, because the random variable X_l^I is always greater than or equal to 0. We can calculate the density function for Y_l^I, denoted by $f_Y^I(y)$, from the density functions for $R_{k,l}^I$. We normalize these density functions with respect to σ_I, so that the probability of error is expressed in terms of E_b/N_0 and E_b/N_I through the parameter v_I given in (1). That is, we evaluate the density functions for $R_{k,l}^I/\sigma_I$, denoted by $f_k^I(x)$. Since we are assuming that the partial-band interference is Gaussian, we may write

$$f_0^I(x) = x \exp\{-\frac{x^2 + v_I^2}{2}\} I_0(x v_I), \quad x \geq 0 \tag{9a}$$

and

$$f_1^I(x) = x \exp\{-\frac{x^2}{2}\}, \quad x \geq 0, \tag{9b}$$

where $I_0(x)$ is the 0-th order modified Bessel function. The density function for Y_l^I, denoted by $f_Y^I(y)$, is

$$f_Y^I(y) = \begin{cases} \displaystyle\int_0^\infty u f_0^I((1+y)u) f_1^I(u)\, du, & -1 \leq y \leq 0 \\ \displaystyle\int_0^\infty u f_0^I(u) f_1^I((1-y)u)\, du, & 0 \leq y \leq 1. \end{cases} \tag{10}$$

The density function for $W^I \triangleq \sum_{l=1}^{j} Y_l^I$, denoted by $f_W^I(y)$ is the $(j-1)$-fold self convolution of $f_Y^I(y)$. (An r-fold self-convolution of f is $\delta(\cdot)$ if $r = -1$, it is f if $r = 0$, it is $f * f$ if $r = 1$, etc.)

Now, consider the diversity receptions with interference absent. Let $f_k^N(x)$ denote the density functions for $R_{k,l}^N/\sigma_I$, and let $X_l^N = R_{0,l}^N/R_{1,l}^N$. Then, the density function for $Y_l^N = T_{0,l}^N - T_{1,l}^N$, denoted by $f_Y^N(y)$, is given by (10), except that we replace $f_k^I(x)$ by $f_k^N(x)$, where

$$f_0^N(x) = \frac{xv_N^2}{v_I^2}\exp\{-\frac{(xv_N/v_I)^2 + v_N^2}{2}\}I_0(\frac{xv_N^2}{v_I}), \quad x \geq 0 \tag{11a}$$

and

$$f_1^N(x) = \frac{xv_N^2}{v_I^2}\exp\{-\frac{(xv_N)^2}{2v_I^2}\}, \quad x \geq 0. \tag{11b}$$

The ratio v_N/v_I appears in the density functions because of the normalization with respect to σ_I. The conditional density function for $W^N \triangleq \sum_{l=j+1}^{L} Y_l^N$, denoted by $f_W^N(y)$, is the $(L-j-1)$-fold self-convolution of $f_Y^N(y)$. The conditional density function for W given that j diversity receptions have interference, denoted by $f_W(y)$, is the convolution of $f_W^I(y)$ with $f_W^N(y)$. The conditional probability of error $P(e;j)$ is the integral from $-\infty$ to 0 of $f_W(y)$.

NUMERICAL RESULTS

For each value of the partial-band interference duty factor ρ, we compute the signal-to-noise ratio required to achieve a given bit error probability. We use (6) to calculate these numerical results for ratio-statistic combining, and we use the probability of error expressions in [4, 6] to obtain results for clipped-linear combining and for the optimum diversity combining scheme. The maximum signal-to-noise ratio requirement over the range of ρ is one system performance measure of interest. We are also interested in the system's narrowband interference rejection capability for Gaussian partial-band interference. This is measured in terms of $\rho_{\min}$ defined as the number such that for all ρ less than $\rho_{\min}$, a given bit error probability can be achieved for Gaussian partial-band interference.

Figure 2 gives curves for E_b/N_I versus the partial-band interference duty factor for receivers with ratio-statistic combining and diversity levels 1-3, and 5. The parameter $\rho_{\min}$ for each diversity level is the value of ρ for which each curve for E_b/N_I versus ρ falls off towards $-\infty$. The figure shows that the narrowband interference rejection capability improves as L increases, at the expense of a higher signal-to-noise ratio requirement for larger values of ρ.

Figure 3 compares ratio-statistic combining to clipped-linear combining. Also, the performance curve for the optimum combining technique for a receiver with perfect side information is shown. This curve is the lower bound on the performance of any diversity combining technique. Although ratio-statistic combining does not perform as well as clipped-linear combining in terms of narrowband interference rejection capability, it does do well. Also, ratio-statistic combining performs better than clipped-linear combining in terms of the signal-to-noise ratio requirement at larger ρ.

It is important to note that the performance of clipped-linear combining changes with the relative clipping level c. A sensitivity analysis of clipped-linear combining is given in [4]. As the relative clipping level increases, the narrowband interference rejection capability of clipped-linear combining decreases. If accurate measurements of the signal output voltage can be obtained, then clipped-linear combining is a useful scheme to employ. For systems in which it is difficult to measure the signal output voltage, the relative clipping level may change from the intended value. For example, in frequency-hop radios, the signal output voltage may vary as the signal is hopped across a wide frequency band. There is also a problem for fading channels; the signal strength of any given diversity reception may vary significantly from the root-mean-squared value. Thus, it is possible that the system will not perform according to the desired specifications due to the sensitivity of clipped-linear combining to c. Diversity transmission systems which employ ratio statistics do not rely on accurate measurements of parameters that are sensitive to fluctuations in the communication channel. Ratio-statistic combining still works under the conditions that make clipped-linear combining impractical.

Figure 4 compares ratio-statistic combining to the optimum combining technique for a channel with partial-band interference and Rayleigh fading. The quantity β_I^2 denotes the average received signal-to-noise ratio given that interference is present. That is, $\beta_I^2 = 2\sigma^2 E_b/N_I$, where $2\sigma^2$ denotes the mean-squared signal strength of the input to the receiver. The term β_N^2 denotes the average received signal-to-noise ratio given that interference is absent, and is equal to $2\sigma^2 E_b/N_0$. For the particular example shown, β_N^2 is set at 22dB to generate the ratio-statistic combining performance curve, and β_N^2 is ∞ for the optimum combining curve. There is a large gap between the performance curve of ratio-statistic combining and the optimum performance curve. However, ratio-statistic combining is still quite effective against partial-band interference and fading. In fact, ratio-statistic combining gives better performance than any of the diversity combining schemes employing ratio statistics analyzed in [6].

CONCLUSIONS

Ratio-statistic combining was analyzed and compared to clipped-linear combining for receivers with no side information in the presence of Gaussian partial-band interference and for general interference. The predicted performance of both techniques in general interference is identical. The performances of ratio-statistic combining and clipped-linear combining in the presence of partial-band interference are very close. When taking into account implementation considerations as well as performance curves, ratio-statistic combining is better than clipped-linear combining. Ratio-statistic combining should be considered as a practical and valuable technique to employ for channels with partial-band interference.

REFERENCES

[1] M. B. Pursley, "Coding and diversity for channels with fading and pulsed interference," *Proc. 16th Conf. Inform. Sci. Syst.,* Princeton Univ., pp. 413-418, Mar. 1982.

[2] M. B. Pursley and W. E. Stark, "Performance of Reed-Solomon coded frequency-hop spread-spectrum communications in partial-band interference," *IEEE Trans. on Commun.,* vol. COM-33, pp. 767-774, Aug. 1985.

[3] W. E. Stark, "Coding for frequency-hopped spread-spectrum communication with partial-band interference—Part I: Capacity and cutoff rate, and Part II: Coded performance," *IEEE Trans. on Commun.,* vol. COM-33, pp. 1036-1057, Oct. 1985.

[4] C. M. Keller and M. B. Pursley, "Diversity combining for frequency-hop spread-spectrum communications with partial-band interference," *IEEE Milit. Commun. Conf.,* pp. 32.1.1-5, Oct. 1984.

[5] C. M. Keller and M. B. Pursley, "A comparison of diversity combining techniques for frequency-hop communications with partial-band interference," *IEEE Milit. Commun. Conf.,* pp. 570-574, Oct. 1985.

[6] C. M. Keller and M. B. Pursley, "Diversity combining for channels with fading and partial-band interference," accepted for publication in the *IEEE J. Select. Areas Commun.,* Apr. 1986, to appear 1987.

[7] J. S. Lee, R. H. French, and L. E. Miller, "Probability of error analyses of a BFSK frequency-hopping system with diversity under partial-band jamming interference—Part I: Performance of square-law linear combining soft decision receivers," *IEEE Trans. Commun.,* vol. COM-32, pp. 645-653, Jun. 1984.

[8] J. S. Lee, L. E. Miller, and Y. K. Kim, "Probability of error analyses of a BFSK frequency-hopping system with diversity under partial-band jamming interference—Part II: Performance of square-law nonlinear combining soft decision receivers," *IEEE Trans. Commun.,* vol. COM-32, pp. 1243-1250, Dec. 1984.

[9] L. E. Miller, J. S. Lee, and A. P. Kadrichu, "Probability of error analyses of a BFSK frequency-hopping system with diversity under partial-band jamming interference—Part III: Performance of a square-law self-normalizing soft decision receiver," *IEEE Trans. Commun.,* vol. COM-34, pp. 669-675, Jul. 1986.

[10] A. J. Viterbi, "A robust ratio-threshold technique to mitigate tone and partial band jamming in coded MFSK systems," *Proc. IEEE Milit. Commun. Conf.,* pp. 22.4-1-5, Oct. 1982.

[11] L. F. Chang and R. J. McEliece, "A study of Viterbi's ratio-threshold AJ technique," *Proc. IEEE Milit. Commun. Conf.,* pp. 11.2.1-5, Oct. 1984.

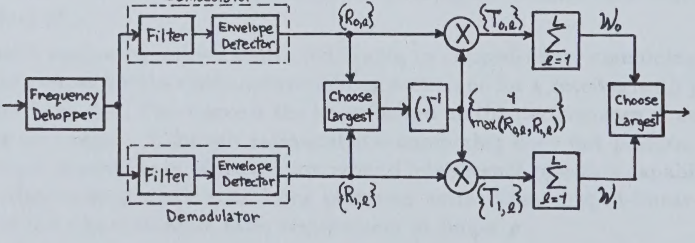


Figure 1. Block diagram for a receiver employing ratio-statistic combining.

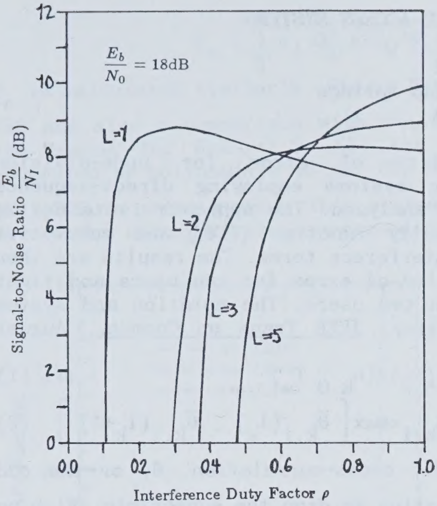


Figure 2. Signal-to-noise ratio required to achieve $p_b = 0.05$ for a receiver with ratio-statistic combining and Gaussian partial-band interference.

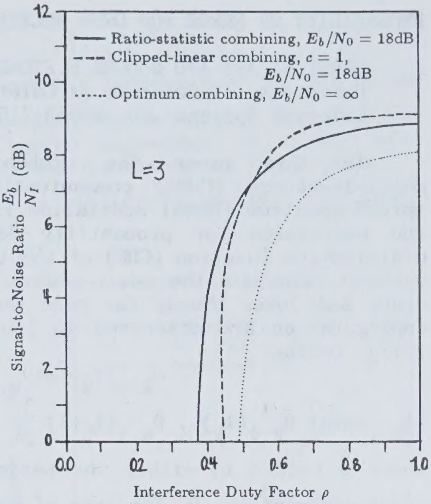


Figure 3. Signal-to-noise ratio required to achieve $p_b = 0.05$ for receivers with Gaussian partial-band interference.

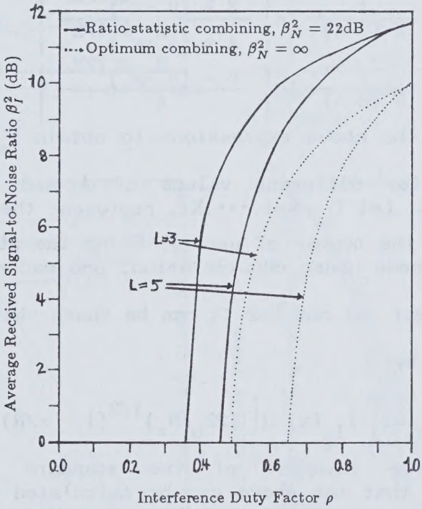


Figure 4. Average received signal-to-noise ratio required to achieve $p_b = 0.05$ for receivers with Rayleigh fading and Gaussian partial-band interference.

PROBABILITY OF ERROR FOR DSSS MULTIPLE-ACCESS SYSTEMS

NERSI NAZARI AND RODGER E.ZIEMER
University of Colorado at Colorado Springs
Colorado Springs, CO 80933-7150

In this paper the probability of error for code-division multiple-access (CDMA) communication systems employing direct-sequence spread-spectrum (DSSS) modulation is analyzed. The approach is to derive the expression for probability density function (PDF) and cumulative distribution function (CDF) of the interferece terms. The results are then used to calculate the exact probability of error for two users and tight upper and lower bounds for more than two users. The notation and system configuration are discussed in [Pursley, IEEE Trans on Commun., August 1977]. Define:

$$m_{k,i}(\tau_k) = v_{k,i}(\tau_k)/(b_{k,0}\, T_c) \tag{1}$$

$$A_{k,i}=\min\left[\bar{\theta}_{k,i}(l_k)\ ,\ \bar{\theta}_{k,i}(l_k+1)\right],\ B_{k,i}=\max\left[\bar{\theta}_{k,i}(l_k)\ ,\ \bar{\theta}_{k,i}(l_k+1)\right] \tag{2}$$

where $\bar{\theta}$ refers to either the periodic cross-correlation, θ, or the odd cross-correlation, $\hat{\theta}$. For ease of notation we drop the subscripts. With no loss of generality one can assume $B>0$ and $|B|>|A|$. The PDF is zero for $|x|>B$. It can then be shown that if $A=B$, the PDF is

$$f_{m|L_k}(x|l_k) = (\pi\ B)^{-1}\left[1-(x/B)^2\right]^{-1/2} \qquad |x|<B \tag{3}$$

If $A \neq B$ we then have

$$f_{m|L_k}(x|l_k) = \frac{1}{\pi\ (B-A)}\ \ln\left|\frac{B+(B^2-x^2)^{1/2}}{A+(A^2-x^2)^{1/2}}\right| \qquad |x|<|A|$$

$$f_{m|L_k}(x|l_k) = \frac{1}{\pi\ (B-A)}\ \ln\left|\frac{B+(B^2-x^2)^{1/2}}{x}\right| \qquad |A|<|x|<B \tag{4}$$

One can integrate the above expressions to obtain $F_{m|L_k}(x|l_k)$. Figure 1 shows the PDF's for different values of A and B; Figure 2 shows corresponding CDF's. Let I_k, $k=1,\cdots,K$, represent the interference due to user k ($k\neq i$, K is the number of users). Using the above results and the statistics of the code under consideration, one can calculate $f_{I_k}(x)$ and $F_{I_k}(x)$. Assuming that $i=1$ and $K=2$ it can be shown that the probability of error, P_e is given by:

$$P_e = \int_{-\infty}^{\infty} f_{I_2}(x)\ Q\left[\ (2E_b/N_o)^{1/2}(1-x/N)\right] \tag{5}$$

where Q(.) is the integral of the standard normal probability distribution. Note that the above can be calculated exactly. For $K>2$ we calculate $\bar{p}_k$ ($\underline{p}_k$), an upper (lower) bound on the probability of total interference to be in interval $i_j=[\underline{x}_j,\bar{x}_j]$. Then, since the Q function is decreasing, the upper (lower) bound for P_e, $\bar{P}_e$ ($\underline{P}_e$) is calculated by:

Work performed under ONR contract N00014-86-R-0555.

$$\bar{P}_e = \sum_j \bar{p}_j \, Q\left[(2E_b/N_o)^{1/2} \, (1 - \bar{x}_k/N) \right] \qquad (6)$$

$\underline{P}_e$ is calculated similarly. Figure 3 shows the tightness of the bounds for K=2 and also a comparison with the Gaussian approximation. Figure 4 has the bounds for K=2,•••,5. All the figures correspond to a Gold code generated by polynomials 45 and 75, N=31 and the number of intervals i_j is (K-1)220 in these graphs. This approach is similar to one used by Lehnert for random codes[private correspondence].

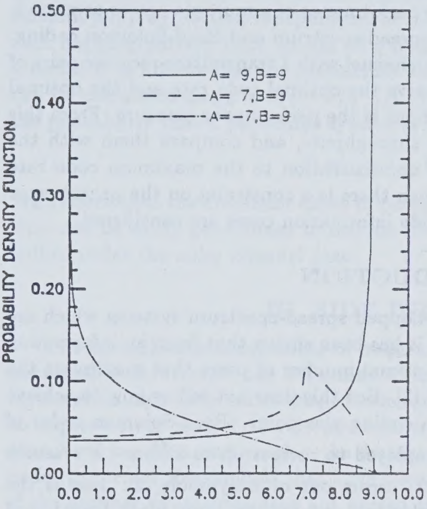


Figure 1. X

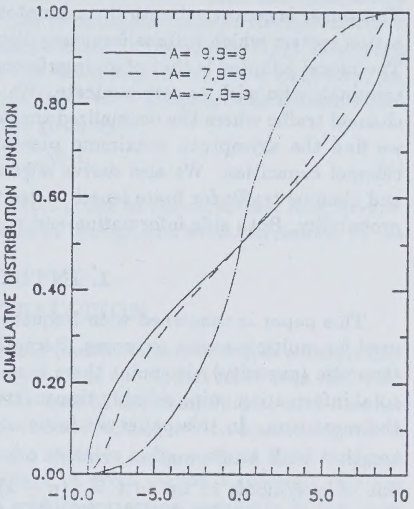


Figure 2. X

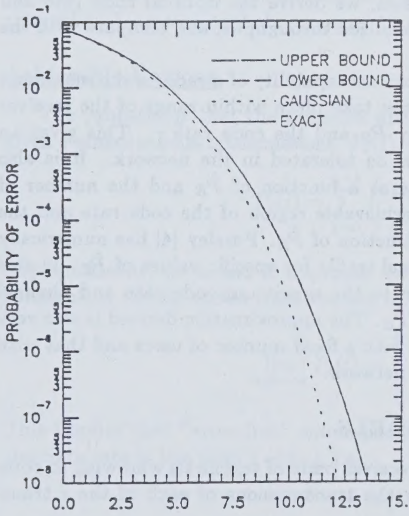


Figure 3. SNR (DB)

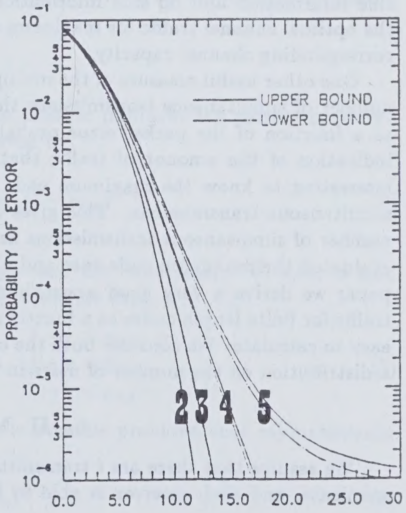


Figure 4. SNR (DB)

MULTIPLE-ACCESS CAPABILITY OF REED-SOLOMON CODED FREQUENCY-HOP SPREAD-SPECTRUM COMMUNICATION SYSTEMS

BAL G. KIM and WAYNE E. STARK
Dept. of Elec.Eng. and Comp.Sci.
Univ. of Michigan, Ann Arbor, MI 48109

ABSTRACT

This paper is concerned with the asymptotic performance of a multiple-access communication system which utilizes frequency-hop spread-spectrum and Reed-Solomon coding. The model adopted is that of an interference channel with i transmitter-receiver pairs of terminals who wish to communicate. We derive the optimal code rate and the optimal channel traffic where the normalized throughput is the performance measure. From this we find the asymptotic maximum possible throughputs, and compare them with the channel capacities. We also derive a good approximation to the maximum code rate and channel traffic for finite length codes when there is a constraint on the packet error probability. Both side information and no side information cases are considered.

I. INTRODUCTION

This paper is concerned with frequency-hopped spread-spectrum systems which are used for multiple-access purposes. Recently it has been shown that from an information theoretic (capacity) viewpoint there is an optimal number of users that maximizes the total information being reliably transmitted [3]. But this does not tell us how to achieve the maximum. In this paper we focus on a coding viewpoint. Reed-Solomon codes of length n with k information symbols are employed to correct up to $e \triangleq n - k$ erasures out of n symbols or up to $t \triangleq \lfloor (n-k)/2 \rfloor$ errors out of n symbols [2]: this is the best value of e and t that can be achieved by any code of the same block length and rate. This is often a strong justification for using Reed-Solomon codes. For both the side information and no side information cases, we derive the optimal code rate and the optimal channel traffic for maximum normalized throughput, and compare with the corresponding channel capacity.

One other useful measure of the multiple-access capability of a radio is the maximum number of simultaneous transmissions that can take place within range of the receiver as a function of the packet error probability $\hat{P}_E$ and the code rate r. This gives an indication of the amount of traffic that can be tolerated in the network. It is also interesting to know the maximum code rate as a function of $\hat{P}_E$ and the number of simultaneous transmissions. This gives an achievable region of the code rate and the number of simultaneous transmissions as a function of $\hat{P}_E$. Pursley [4] has numerically evaluated the maximum code rate and channel traffic for specific values of $\hat{P}_E$. In this paper we derive a very good approximation to the maximum code rate and channel traffic for finite length codes as a function of $\hat{P}_E$. The approximation derived is also very easy to calculate. We consider both the case with a fixed number of users and that with a distribution on the number of users in the network.

II. MODELS

We assume that there are i transmitter-receiver pairs of terminals who wish to communicate, and each receiver is able to hear the transmissions of each of the i transmitters. An example of this is a satellite multiple-access system or a fully connected network. Each packet consists of one codeword and one symbol is transmitted for each hop. The output of the (n, k) Reed-Solomon encoder is a block of n symbols from an M-ary alphabet. For the codes considered in this paper n and M are equal, and the bounded distance decoding is employed to correct erasures or errors. For most radio networks, synchronous frequency hopping is not feasible. In fact, if the differential propagation times are larger than the hop interval duration, it is physically impossible for the hop intervals to be aligned at all of the receivers in the network. Consequently, this paper is concerned with asynchronous frequency-hop radio network. However, we assume that synchronization is maintained at the packet level. Each user pair has a random hopping pattern independent of all other hopping patterns in the network and each hopping pattern hops among all q frequency slots with probability $1/q$. With such a model it is known that the probability that two users hop to the same frequency, p_h, can be calculated as $p_h = 2/q - 1/q^2$ [1]. When there are i simultaneous transmissions the probability that a particular symbol is hit is given by

$$p_{h,i} = 1 - (1 - p_h)^{i-1}. \tag{1}$$

We assume that the dominant cause of packet errors is due to the multi-user interference. This can be easily generalized to include the case of background noise. In section IV we will consider the noisy channel case.

III. SIDE INFORMATION

The multiple-access capability of frequency-hop packet radio networks is considerably enhanced if the receiver can determine which symbols in the packet have been hit. This information, typically called side information, is used in the demodulator and decoder to improve error control. A very powerful error-control technique is to erase the symbols that have been hit and employ the features of a suitable code, such as Reed-Solomon code, to correct the symbols that have been erased. A given packet is correctly received if the number of erasures does not exceed the erasure-correction capability of the code: for (n, k) Reed-Solomon code it is $n - k$. We assume that hits are detected and the symbols hit are erased. The induced channel model is thus an M-ary symmetric erasure channel with erasure probability equal to $p_{h,i}$.

Achievable Regions

The probability that the bounded distance decoder produces the correct codeword given i simultaneous transmissions, $P_c(i)$, is given by

$$P_c(i) = \sum_{j=0}^{n-k} \binom{n}{j} p_{h,i}^{j} (1 - p_{h,i})^{n-j}. \tag{2}$$

Asymptotically, as n and k approach infinity while the code rate, $r \triangleq k/n$, is held constant, it can be shown from the law of large numbers that

$$\lim_{n,k\to\infty} P_c(i) = \begin{cases} 1, & 1 - r > p_{h,i} \\ 0.5, & 1 - r = p_{h,i} \\ 0, & 1 - r < p_{h,i}. \end{cases} \tag{3}$$

This implies that "error-free" communication is possible provided that asymptotically the code rate is less than $1 - p_{h,i}$, i.e.,

$$r < (1 - p_h)^{i-1}, \tag{4}$$

or equivalently

$$i < 1 + ln(r)/ln(1 - p_h). \tag{5}$$

This represents an asymptotic achievable region of i and r for arbitrarily small error probability with side information.

Now we consider the achievable region with finite block length Reed-Solomon codes. We want to find the maximum number of simultaneous transmissions(or channel traffic) and the maximum code rate for the packet error probability to be less than $\hat{P}_E$ when Reed-Solomon codes of finite block length are used to correct erasures due to the multi-user interferences. Unfortunately it is very difficult to derive the exact maximums analytically, because (2) is a complicated function of i and k. However, we can derive a simple approximation to the maximum channel traffic and code rate as follows. If we apply the Demoivre-Laplace theorem [6] to (2), we can get an approximation

$$P_c(i) \approx F\left(\frac{n - k - np_{h,i}}{\sqrt{np_{h,i}(1 - p_{h,i})}}\right), \tag{6}$$

where

$$F(x) \triangleq \int_{-\infty}^{x} (2\pi)^{-\frac{1}{2}} e^{-u^2/2} du.$$

By combining (6) with the constraint, $P_E(i) \triangleq 1 - P_c(i) \leq \hat{P}_E$, we obtain

$$r \leq 1 - p_{h,i} - \alpha\sqrt{p_{h,i}(1 - p_{h,i})/n}, \text{ or } i \leq 1 + ln(A)/ln(1 - p_h), \tag{7}$$

where

$$\alpha \triangleq F^{-1}(1 - \hat{P}_E),\ A \triangleq \frac{2nr + \alpha^2 + \alpha\sqrt{4nr(1 - r) + \alpha^2}}{2(n + \alpha^2)}.$$

Fig.1 shows the achievable regions of i and r for $P_E(i) \leq 10^{-2}$ with length 32 and 256 Reed-Solomon codes and also the asymptotic achievable regions for arbitrarily small error probability with side information. We see that the exact values and the approximations are very close, where the exact values have been obtained by computer search.

When there is a distribution on the number of packet transmissions (Possion with mean G), we have the average probability of correct decoding, P_C, given by

$$P_C = \sum_{i=0}^{\infty} P_c(i + 1)\, \frac{G^i e^{-G}}{i\,!}., \tag{8}$$

It is shown in [4] that

$$P_C = \sum_{j=0}^{n-k} \binom{n}{j} (1 - e^{-2\mu})^j (e^{-2\mu})^{n-j}, \tag{9}$$

as G and q approaches infinity while $\mu \triangleq G/q$ is held constant. To satisfy the constraint, $P_E \triangleq 1 - P_C \leq \hat{P}_E$, we must have

$$r \leq e^{-2\mu} - \alpha\sqrt{e^{-2\mu}(1 - e^{-2\mu})/n}, \text{ or } \mu \leq \frac{1}{2} ln(1/A). \tag{10}$$

Throughput

For a system employing (n, k) code and q frequency slots, the normalized throughput given i simultaneous transmissions is given by

$$W = \frac{k}{nq} i P_c(i). \tag{11}$$

The asymptotic normalized throughput as n and k approach infinity while $r \triangleq k/n$ is held constant can be obtained by using (5) and (11) as

$$\lim_{n,k\to\infty} W = \frac{r}{q}\left(1 + \frac{ln(r)}{ln(1-p_h)}\right). \tag{12}$$

The unit of the above normalized throughput is information symbols per channel use per time slot. To change it to information bits per dimension per time slot, it must be multiplied by $log_2 M/D$, where D is the dimensionality of the signal set. In all that follows we consider the former unit.

By optimizing over r, we get the optimum code rate, r_{opt}, given by

$$r_{opt} = e^{-1}(1-p_h)^{-1}, \tag{13}$$

which for reasonable values of q is very close to e^{-1}. This asymptotic optimal code rate also seems to give a very good approximation for finite length codes, even the length 32 codes: the maximum possible throughput is obtained by the (32,12) rate 0.375 code and (256,94) rate 0.367 code when q=100 (see Fig.2). It can also be shown that the optimum value of the traffic intensity, $\lambda \triangleq i/q$, is $\lambda_{opt} = 0.5$, as $i, q \to \infty$, while λ is held constant. At r_{opt} and λ_{opt} the maximum possible throughput, W_{max} becomes

$$W_{max} = 1/[qe(1-p_h)ln(1-p_h)^{-1}], \tag{14}$$

which approaches $e^{-1}/2$ for large enough q. This is also the asymptotic sum capacity in information symbols per channel use [3],[5]. Thus if side information is available, the channel capacity is achieved by the optimal rate Reed-Solomon code with bounded distance decoding. However, this is not true if side information is not available as we will see in next section.

IV. NO SIDE INFORMATION

The demodulator no longer produces the erasure symbol, but instead makes a maximum likelihood estimate on the received signal. Thus the channel model induced is the M-ary symmetric error channel. We assume first that if a received symbol is hit, then the demodulator output is equally likely one of the M possible symbols. This is the worst case assumption, so the resulting achievable regions and throughput are pessimistic results. The conditional probability of symbol error given hit P_1 is thus given by $1 - 1/M$. We assume for the moment that the signal power is high enough to ignore the effect of background noise. Then the probability of a received symbol being in error given i simultaneous transmissions, $p_{e,i}$, is

$$p_{e,i} = (1 - 1/M)(1 - (1-p_h)^{i-1}). \tag{15}$$

Achievable Regions

When (n, k) Reed-Solomon code is employed to correct the errors, the probability of correctly decoding a packet given i simultaneous transmissions, $P_c(i)$, with bounded distance decoding is given by

$$P_c(i) = \sum_{j=0}^{\lfloor (n-k)/2 \rfloor} \binom{n}{j} p_{e,i}^j (1-p_{e,i})^{n-j}. \tag{16}$$

Asymptotically, as n and k approach infinity while the code rate, $r \triangleq k/n$, is held constant, it can be shown from the law of large numbers that

$$\lim_{n,k\to\infty} P_c(i) = \begin{cases} 1, & 1-r > 2p_{e,i} \\ 0.5, & 1-r = 2p_{e,i} \\ 0, & 1-r < 2p_{e,i}. \end{cases} \tag{17}$$

This implies that "error-free" communication is possible provided that asymptotically the code rate is less than $1-2p_{e,i}$, i.e.,

$$r < 2(1-p_h)^{i-1} - 1, \tag{18}$$

or equivalently

$$i < 1 + ln((1+r)/2)/ln(1-p_h). \tag{19}$$

This represents an asymptotic achievable region of i and r for arbitrarily small error probability without side information.

Also for the case of finite block length codes, we can derive a similar approximation to the maximum channel traffic and code rate for the packet error probability to be less than $\hat{P}_E$. By applying the Demoivre-Laplace theorem to (16), we can get an approximation

$$P_c(i) \approx F\left(\frac{\lfloor (n-k)/2 \rfloor - np_{e,i}}{\sqrt{np_{e,i}(1-p_{e,i})}} \right). \tag{20}$$

Combining (20) with the constraint, $P_E(i) \triangleq 1 - P_c(i) \le \hat{P}_E$, we have

$$r \le 1 - \frac{2}{n}\lceil np_{e,i} + \alpha\sqrt{np_{e,i}(1-p_{e,i})}\rceil, \text{ or } i \le 1 + ln(B)/ln(1-p_h), \tag{21}$$

where

$$B \triangleq 1 - \frac{2t + \alpha^2 - \alpha\sqrt{4t + \alpha^2 - 4t^2/n}}{2(1-1/M)(n+\alpha^2)}$$

Fig.3 shows the achievable regions of i and r for $P_E(i) \le 10^{-2}$ with length 32 and 256 Reed-Solomon codes and also the asymptotic achievable regions for arbitrarily small error probability without side information. We see that the exact values and the approximations are very close.

When there is a distribution on the number of packet transmissions, it can be shown that

$$P_C = \sum_{j=0}^{t} \binom{n}{j} [(1-1/M)(1-e^{-2\mu})]^j [1-(1-1/M)(1-e^{-2\mu})]^{n-j}, \tag{22}$$

as G and q approach infinity while $\mu \triangleq G/q$ is held constant. To satisfy the constraint, $P_E \le \hat{P}_E$, we must have

$$r \le 1 - \frac{2}{n}\lceil np_{e,\mu} + \alpha\sqrt{np_{e,\mu}(1-p_{e,\mu})}\rceil, \text{ or } \mu \le \frac{1}{2}ln(1/B), \tag{23}$$

where

$$p_{e,\mu} \triangleq (1-1/M)(1-e^{-2\mu}).$$

Throughput

As in side information case, the asymptotic normalized throughput can be obtained by using (11) and (19) as

$$\lim_{n,k \to \infty} W = \frac{r}{q}\left(1 + \frac{ln((1+r)/2)}{ln(1-p_h)}\right), \tag{24}$$

which approaches $\frac{r}{2} ln(\frac{2}{1+r})$ for large enough q. By optimizing over r we get the optimal code rate, r_{opt}

$$r_{opt} = 0.4597, \tag{25}$$

and the optimum traffic intensity, $\lambda_{opt} = 0.1574$, and the asymptotic maximum possible throughput, $W_{max} = 0.0724$. This asymptotic optimal code rate also seems to give a very good approximation for finite length codes, even the length 32 codes: the maximum possible throughput is obtained by the (32,14) rate 0.437 code and (256,118) rate 0.460 code when q=100. In addition, the asymptotic λ_{opt} also seems to give a good approximation for finite q (see Fig.4). However it can be shown that the asymptotic($M, q \to \infty$) sum capacity, C_{sum} approaches $e^{-1}/2$, which is the same as in side information case. The loss in using Reed-Solomon codes is partially due to the use of a non maximum likelihood decoder (i.e., bounded distance decoding).

So far, we have ignored the effects of channel background noise. This can be incorporated as follows. If the conditional probability of symbol error given that no hits have occured is P_o, then the symbol error probability given i transmissions, $p_{e,i}$, is given by

$$p_{e,i} = (1 - 1/M)(1 - (1-p_h)^{i-1}) + P_o(1-p_h)^{i-1}. \tag{26}$$

If (26) instead of (15) is applied to (21), we can obtain the maximum code rate on the noisy channel. Similarly, we can obtain the maximum number of simultaneous transmissions on the noisy channel, which is given by

$$i \leq 1 + ln(D)/ln(1-p_h), \tag{27}$$

where

$$D \triangleq \frac{\frac{2t+\alpha^2-\alpha\sqrt{4t-4t^2/n+\alpha^2}}{2(n+\alpha^2)} - \frac{M-1}{M}}{P_o - \frac{M-1}{M}}.$$

As an example consider an M-ary signal set formed from binary frequency-shift key(BFSK) signals of length $log_2 M$. For this signal set used over an additive white Gaussian noise channel, $P_o = 1 - (1 - \frac{1}{2}e^{-E/2N_0})^{log_2 M}$, where E is the energy in the received signal and $N_0/2$ is the two-sided spectral density of the noise. In Fig.5 the maximum code rate and the maximum number of simultaneous transmissions are plotted in terms of the signal to noise ratios. We can see that the exact values and the approximations are very close, and if signal to noise ratio is greater than threshold the effect of background noise is negligible.

Consider now M-ary orthogonal signaling. Notice that we have not distinguished one hit from more than one hit in the calculation of above $p_{e,i}$. This can be taken into account as follows. If two users hop to the same frequency bin at the same time, i.e. one hit, the probability of symbol error is 1/2 if the two users have different symbol and 0 if they have the same symbol. If three users collide, i.e. two hits, the symbol error probability is 2/3 if all three users transmit different symbol, and 1/2 if two of the users transmit the same symbol but the third transmits a different symbol, and 0 if they have the same symbol. The probability of symbol error with m hits, P_m, can be derived as:

$$\begin{aligned}
P_1 &= 0\tfrac{1}{M} + \tfrac{1}{2}(1 - \tfrac{1}{M}) = \tfrac{1}{2}(\tfrac{M-1}{M}) \\
P_2 &= 0\tfrac{1}{M^2} + \tfrac{1}{2}\tfrac{3(M-1)}{M^2} + \tfrac{2}{3}\tfrac{(M-1)(M-2)}{M^2} = \tfrac{4M^2-3M-1}{6M^2} \\
P_3 &= \tfrac{3M^3-2M^2-M}{4M^3} \\
P_4 &= \tfrac{24M^4-15M^3-10M^2+1}{30M^4} \\
P_5 &= \tfrac{10M^5-6M^4-5M^3+M}{12M^5}. \\
&\;\;\vdots
\end{aligned} \tag{28}$$

Thus for large enough M, P_m is given by $P_m = m/(m+1)$. This is also an upper bound for finite M, and can also be used as a good approximation: for example, P_4=0.784 for M=32 and 0.798 for M=256, while the upper bound is 0.8. By averaging over the number of hits we can determine the average symbol error probability, $p_{e,i}$,

$$\begin{aligned}
p_{e,i} &= \textstyle\sum_{m=1}^{i-1} \frac{m}{m+1} \begin{pmatrix} i-1 \\ m \end{pmatrix} p_h^m (1-p_h)^{i-1-m} \\
&= 1 - \tfrac{1-(1-p_h)^i}{ip_h}.
\end{aligned} \tag{29}$$

Achievable Regions

From (17) and (29) we can show that asymptotically as $n, k \to \infty$ while $r \triangleq k/n$ is held constant, "error-free" communication is possible, i.e, $P_c(i)$=1, provided

$$r < \frac{2 - 2(1-p_h)^i}{ip_h} - 1. \tag{30}$$

Throughput

From (11) and (30) the asymptotic normalized throughput can be shown to be

$$\lim_{n,k\to\infty} W = \frac{i}{q}\left(\frac{2-2(1-p_h)^i}{ip_h} - 1\right), \tag{31}$$

which approaches $1 - e^{-2\lambda} - \lambda$ as $i, q \to \infty$ while $\lambda \triangleq i/q$ is held constant. By optimizing over λ we get the optimum traffic intensity, $\lambda_{opt} = ln(2)/2 = 0.3466$, and the optimum code rate, r_{opt}

$$\begin{aligned}
r_{opt} &= \tfrac{1-e^{-2\lambda_{opt}}}{\lambda_{opt}} - 1 \\
&= ln(e/2)/ln(2) = 0.4427,
\end{aligned} \tag{32}$$

and the asymptotic maximum possible throughput, W_{max}

$$W_{max} = ln(e/2)/2 = 0.1534. \tag{33}$$

However the sum capacity region with this model is not known.

Although we have not discussed the hop synchronous system, it can be easily shown that asymptotically W_{max} and λ_{opt} of the hop synchronous system are twice as much as those of the asynchronous system and r_{opt} is the same for both systems.

V. CONCLUSIONS

A multiple-access communication system which utilizes frequency-hop spread-spectrum and Reed-Solomon coding has been investigated. We have derived the optimal code rate and the optimal channel traffic for both side information and no side information cases where the normalized throughput is the performance measure. From this we have derived the asymptotic maximum possible throughputs, and compared them with the channel capacities. For the case of side information, the channel capacity is achieved by the optimal rate Reed-Solomon code with bounded distance decoding. However this is not true if the side information is not available, because the bounded distance decoding is not the optimal decoding for that case. However, we have recently found that the channel capacity for the case of no side information is achieved by concatenating the Reed-Solomon outer code with error detecting inner codes. The performance of concatenated codes on multiple-access channels will be given in a subsequent paper.

One other useful measure of the multiple-access capability of a radio is the maximum number of simultaneous transmissions and the maximum code rate as a function of the packet error probability, $\hat{P}_E$. This gives an achievable region of the channel traffic and code rate as a function of $\hat{P}_E$. We have derived a very good approximation to the achievable region for finite length codes as a function of $\hat{P}_E$ and also the exact asymptotic achievable region for arbitrarily small error probability for both side information and no side information cases. The approximation derived is also very easy to calculate. Both fixed number of users and a distribution on the number of users in the network have been investigated.

ACKNOWLEDGEMENT

This work was supported by the Office of Naval Research under Contract N00014-85-K-0545.

REFERENCES

[1] E.A.Geraniotis and M.B.Pursley, "Error probabilities for slow frequency-hopped spread-spectrum multiple access communications over fading channels," *IEEE Tr. on Comm.*, pp.996-1009,May 1982.

[2] G.C.Clark and J.B.Cain,*Error-Correction Coding for Digital Communications*, New York:Plenum, 1981.

[3] M.Hegde and W.E.Stark, "Multiple-access capability of frequency hop spread-spectrum communication," *Proc. of the 1985 IEEE Milcom*, Oct. 1985.

[4] M.B.Pursley, "Frequency-hop transmission for satellite packet switching and terrestrial packet radio networks," *IEEE Tr.Infor. Th.*, To appear.

[5] B.Hajek, "Adaptive packet radio networks," Oct. 1983.

[6] A.Papoulis, *Probability, Random Variables, and Stochastic Processes*, p.66, McGraw-Hill,1965.

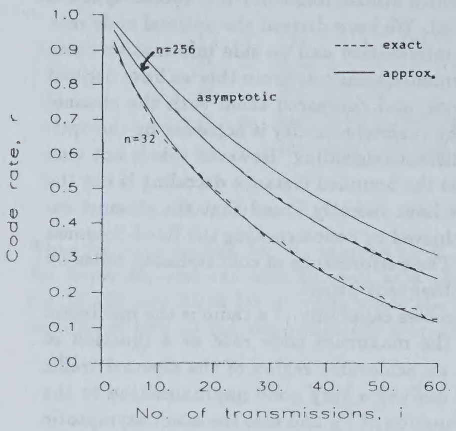


Fig.1. Side information

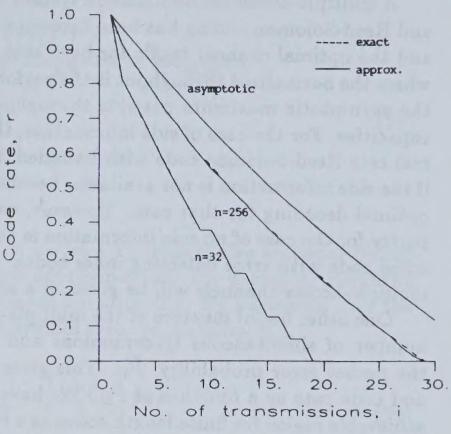


Fig.3. No side information

Achievable regions of the number of simultaneous transmissions and code rate for $P_E(i) \leqslant 10^{-2}$ and the asymptotic achievable region, q=100.

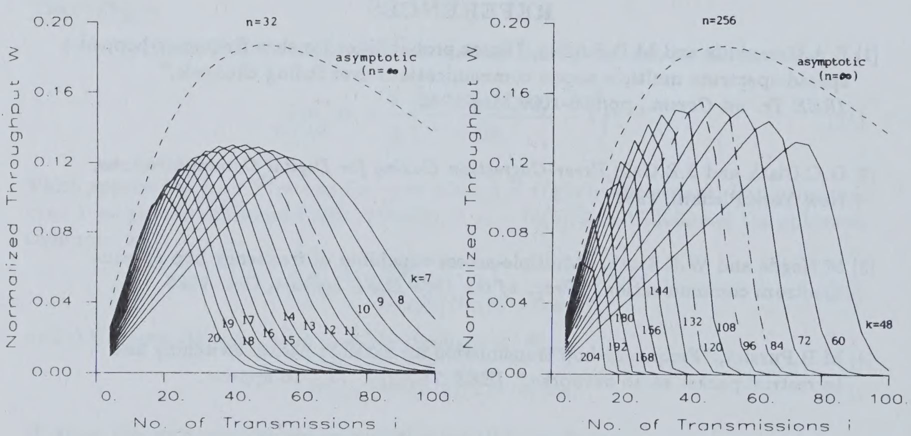


Fig.2. Normalized throughputs, side information, q=100.

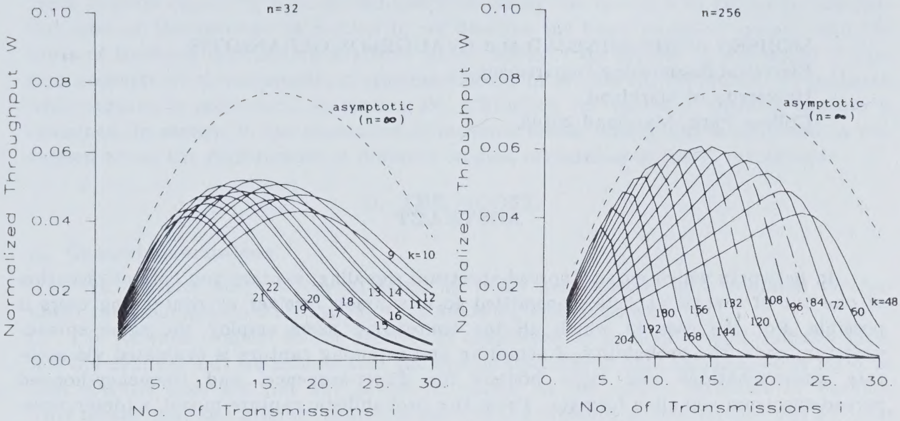


Fig.4. Normalized throughputs, no side information, q=100.

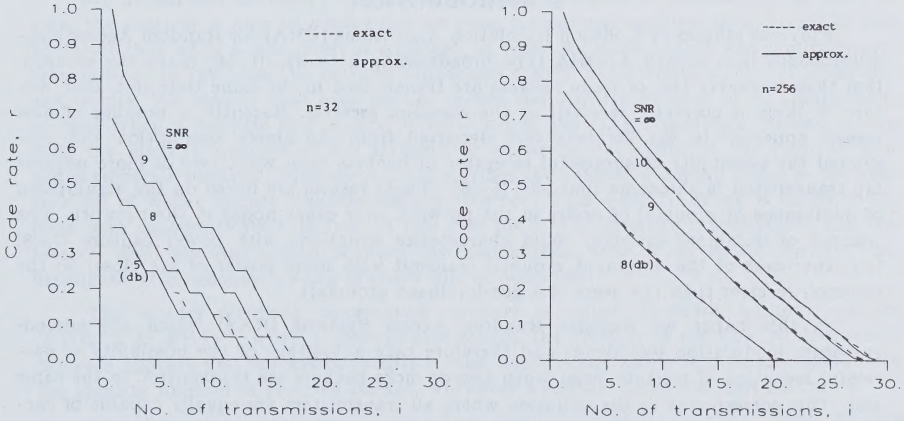


Fig.5. Max. code rate and max. packet transmissions for $P_E(i) \leq 10^{-2}$, no side information, noisy channel, q=100.

COLLISION RESOLUTION ALGORITHMS FOR NETWORKS WITH SPREAD-SPECTRUM CAPTURE CAPABILITY

MOHSEN SOROUSHNEJAD and EVAGGELOS GERANIOTIS
Electrical Engineering Department
University of Maryland
College Park, Maryland 20742

ABSTRACT

In networks which employ spread-spectrum signaling, capture and correct reception of one out of several packets transmitted to a common receiver by contending users is possible. For the case in which all the contending users employ the same spread-spectrum code, the probability of acquiring and retaining capture is evaluated via accurate approximations and tight bounds for direct-sequence, and frequency-hopped spread-spectrum signaling formats. From this probabilistic capture model, a (deterministic) threshold capture model is also derived.

Both these models of capture are then incorporated into the binary tree collision resolution algorithm which takes advantage of the capture capability provided by the spread-spectum signaling. Stable throughputs are evaluated for two types of feedback: (i) feedback with capture (4-ary), in which case the receiver can distinguish between capture and success slots (as well as between idle slots and collision slots as in the ternary feedback non-capture case), and (ii) feedback without capture (ternary) in which case the receiver can not distinguish between capture and success slots.

I. INTRODUCTION

Previous studies of Collision Resolution Algorithms (CRA) for Random Access communications in a slotted ALOHA type broadcasting network, [1]-[4], make the assumption that whenever two or more packets are transmitted in the same time slot, then neither of them is correctly received at the common receiver. Recently, a number of new results appeared in the litrature that departed from the above assumption and considered the possibility of successful reception of packets even when two or more packets are transmitted in the same time slot [5]-[6]. These results are based on the assumption of dominance of group(s) of nodes in the network over other nodes in the network. The concept of dominant group(s) could characterize situations with power capture [7]-[8] (e.g. the users of the dominant group(s) transmit with more power or are closer to the common receiver than the users of a nondominant group(s)).

In this paper we consider Random Access Systems (RAS) which use spread-spectrum modulation techniques and therefore take advantage of the possibility of successful reception of packets, even when two or more packets are transmitted in the same slot. This corresponds to the situation where all transmitters are equally capable of capturing the receiver; this is termed *delay (or code) capture* [8] and is described in section II-B.

In particular, we investigate the basic binary tree CRA of Capetanakis [1], and Tsybakov and Mikhailov [2] (CTMCRA) in a slotted ALOHA type broadcasting network with *code capture* capability. This algorithm is known to be very flexible and insensitive

---

This research was supported by the National Science Foundation through grant ECS-85-16689.

to channel errors [3].

It should be emphasized that our intention is not to make a comparison between conventional RAS's and RAS's with capture capability. Rather, given that our system has a capture capability (i.e., spread-spectrum RAS), the intent is to determine the performance of the system. In section II, we describe the basic model of system, and the types of feedback information available to the nodes of the system. A discussion of capture property of spread-spectrum systems follows. In section III, CTMCRA for feedback with capture is presented. In section IV, CTMCRA for feedback without capture is examined. In section V, the evaluation of maximal stable throughput is discussed. A discussion about the performance of different models, introduced in III-IV, is followed.

II. THE MODEL

A. General assumptions

We consider the random-accessing by many transmitters of a common receiver under the assumptions stated in [3], with following modifications :

(i) The forward channel to the receiver is considered to be time-slotted collision type, AWGN channel, but we assume that the system operates at high signal-to-noise ratio, so that AWGN does not put any severe limitation on the correct reception of packets. This means that the primary source of error is the presence of interference due to other users in the same time slot.

(ii) The possible events that may occur during each slot are as follows: a) Idle slot, b) Success slot, c) Collision slot, and d) Capture slot. See [5] for explanation of these events. If any of the above events a), b), c), or d) occurs, the receiver broadcasts the feedback messages LACK, ACK, NACK, or CAPT respectively.

The basic CTMCRA (with no capture capability) is described in [3]. We consider the channel-access protocol known as *obvious blocked-access* protocol. This means that a transmitter sends a new packet in the first slot following the resolution of all collisions that had occured prior to the arrival of the packet.

We distinguish between two possible feedback information channels [5]. In the first case, the receiver is able to distinguish between success and capture slots, i.e., it is able to detect that while it has received a packet, at least one more packet was simultanously transmitted. In this case the receiver broadcasts CAPT to all nodes whenever a capture slot occurs. We assume that each packet carries the identity number of its transmitter, so that when a capture slot occurs, receiver sends the CAPT feedback message along with the identity number of the transmitter whose packet captured the receiver, and hence correctly received. This assumption clarifies the confusion among nodes in the *capture set* of whose packet was successfully received. This 4-ary feedback is called, feedback with capture (FWC). In fact, in this case the LACK message is redundant, and the receiver need not distinguish between idle and success slots. That is, the receiver can broadcast ACK whenever a slot is either idle or successful.

The second feedback information channel is called, feedback without capture (FWOC). In FWOC, when a capture slot occurs, the receiver is not able to distinguish between capture and successful reception, and hence sends an ACK message, along with the identity number of received packet, whenever a capture or success slot occurs.

Two important performance measures that characterize a CRA (with *blocked -access protocol*) are the average collision resolution interval length, conditioned on the number of packets involved in the initial collision and the maximum attainable throughput. The Collision Resolution Interval (CRI) is defined as the time elapsed from an initial collision (capture) until it is resolved. The maximum attainable throughput is defined as the maximum allowable arrival rate of new packets into the system for which the system is stable.

B. Capture phenomena.

Capture phenomena characterizes the ability of a receiver to successfully receive a packet (with nonzero probability) even though part or all of the packet arrives at the receiver overlapped in time by other packets. The basic mechanism for capture is the ability of the receiver to synchronize with and lock on to one packet and subsequently reject other overlapping packets as noise [9].

Systems which use spread-spectrum modulation techniques may exhibit the *delay capture* phenomenon. When the nodes of a network employ the same spread-spectrum code that does not repeat within a packet duration, then the packets would be strongly correlated over each data symbol if they arrived at a receiver simultanously, but would be pseudo-orthogonal if they arrived with a time offset. In general, there is a vulnerable period at the beginning of a packet, denoted by T_c and called *capture interval,* during which collision with the same portion of another packet results in the loss of both [9].

The probability of capture for the *delay capture* model described above, for different spread-spectrum modulation techniques [i.e., direct-sequence (DS/SS), frequency-hopped (FH/SS), and hybrid (DS-FH/SS)] is derived in our work of [11]. Next, we give a brief description of these derivations.

The probability of capture is defined as

$$P_c(n) \equiv Pr(1 \text{ user captured} \mid n \text{ users contend}) \quad ; \quad n \geq 2.$$

$P_c(n)$ can be decomposed into two parts; the probability of acquisition of capture, $P_a(n)$, and the probability of retaining capture, denoted by $P_r(n)$ (i.e., $P_c(n) = P_a(n)\, P_r(n)$). The probability of acquisition for the *delay capture* has been already derived by Gronemeyer and Davis [10]. By introducing a time of arrival randomization procedure, with parameter T_u, to eliminate discrimination as a function of range, $P_a(n)$ is given by:

$$P_a(n) = \begin{cases} 1 & n = 1 \\ (1 - Q)^n & n \geq 2 \\ 0 & otherwise \end{cases}$$

where $Q \equiv \dfrac{T_c}{T_u}$, is called the capture ratio [10].

The probability of retaining capture pertains to the rejection of the interfering packets as noise, and therefore depends on the SS modulation scheme and correlation properties of the signature sequences or hopping patterns. We model the signature sequence (or hopping pattern) being used by all users, as a random SS code (e.g., random binary sequence for DS/SS and random hopping pattern for FH/SS) and use techniques similar to those used for the evaluation of error probabilities in the case of multiple-access interference to calculate the probability of retaining capture. For DS/SS or hybrid (FH-DS) SS, the exact calculation of this probability is intracable, so we use appropriate approximations; for FH/SS we obtain a tight upper bound. It is shown in [11], that $P_r(n)$ can be written as

$$P_r(n) = (1 - P_{es}(n))^L$$

$P_{es}(n)$ is the probability of symbol error in the presense of $n-1$ interfering users, and L is the number of symbols per packet. In above, we have made the assumption that symbol errors within a packet are independent. This is not true in the DS-SS case, because of strong correlation of symbols, due to the fact that all nodes use the same SS code. In FH/SS case, proper interleaving of symbols would achieve the independence of errors. Figures c1.-c2. present the probability of capture for DS/SS and FH/SS systems with different parameters. See [11] for more detail.

We incoporate the capture effect in the CTMCRA in two forms. According to the first model, if no more than Γ packets transmit in a given slot, then one packet is captured and correctly received, while the others should be retransmitted. If more than Γ nodes contend, there will be a collision event and all of them must be retransmitted. Γ is called the *capture capacity* of the system. We refer to this model as *deterministic capture model* (DCM). The capture parameter, Γ, can be obtained directly from the probability of capture by imposing a hard constraint (upper bound) on the tolerable probability of no capture.

According to the second model, called *the probabilistic capture model* (PCM), with probability $P_c(n)$, one out of n contending packets is captured and correctly received. $P_c(n)$ is a monotone decreasing function of n. This means that for $n > N$, N a positive integer obtained from some performance index, the probability of capture falls below an acceptable level, say P_{cu}. For this reason, we carry out the performance evaluation of the CRA under the assumption that no capture occurs when $n > N$; i.e., set $P_c(n) = 0$ whenever $P_c(n) < P_{cu}$.

III. CTMCRA FOR FEEDBACK WITH CAPTURE

In feedback with capture (FWC), where the receiver can distinguish between capture and success slots, we assume that whenever a capture event occurs all packets belonging to a capture set must be retransmitted and eventually be correctly received at the common receiver before any other packets may be (re)transmitted.

Two possible actions might be taken by nodes from the capture set that are involved in a capture slot. One possibility is to perform the exact CTMCRA, i.e., flip a coin, etc. We shall call this scheme 1. The other possibility is to transmit during the slot immediately following the capture slot and then continue to perform the CTMCRA. We call this scheme 2 [5].

A. DCM - FWC

Let the random variable X denote the number of packets transmitted in the first slot of a CRI, and let Y denote the length of the same CRI. We define Y_n as the (random) length of time required by the CRA to resolve a contention among n nodes. Then the conditional mean of the CRI length is given by

$$L_n \equiv E(Y_n) = E(Y \mid X = n) \tag{3.1}$$

It is easy to see that given X, Y depends only on the results of the coin tosses performed internally in the algorithm, and hence independent of the arrival process of new packets into the system.

First, considering scheme 1, it is obvious that $Y_0 = Y_1 = 1$. When $n \geq 2$, we have two possibilities : i) $2 \leq n \leq \Gamma$; or ii) $n > \Gamma \geq 2$.

In the first case (i), a capture occurs in the first slot of CRI, and the remaining n-1 nodes form a capture set and flip their binary coin.

From Figure 3.1(a), we see that Y_n is given by

$$Y_n = 1 + Y_{S_{n-1}} + Y_{n-S_{n-1}-1} \tag{3.2a}$$

where S_n is a binomially distributed random variable with PMF

$$P(S_n = i) \equiv P_n(i) = \binom{n}{i} 2^{-n} \quad ; \quad 0 \leq i \leq n \tag{3.3}$$

Using (3.2a) and (3.3), we find the conditional mean of the CRI as

$$L_n = 1 + 2 \sum_{i=0}^{n-1} L_i \; P_{n-1}(i) \quad ; \quad 2 \leq n \leq \Gamma \tag{3.2b}$$

where we have used the fact that $P_n(i) = P_n(n-i)$.

In the second case (ii), the first slot of the CRI is a collision slot (see Figure 3.1(b)), and the recursion relation for L_n is given by

$$L_n = [1 + 2 \sum_{i=0}^{n-1} L_i P_n(i)] / [1 - 2 P_n(n)] \quad ; \; n > \Gamma \tag{3.2c}$$

Equation (3.2c) is same as the recursion relation given in [3] for basic CTMCRA. The only difference is that for $2 \leq i \leq \Gamma$, the values of L_i are obtained by the recursion relation of (3.2b).

Regarding scheme 2, we notice that when $2 \leq n \leq \Gamma$, the first slot of CRI is a capture slot, and therefore by rule of scheme 2, all the following slots, except the last one, are capture slots. The last slot of the CRI will be a success slot. This gives us

$$L_n = n \qquad \text{for } 2 \leq n \leq \Gamma. \tag{3.4}$$

When $n > \Gamma$, recursion relation of (3.2c) holds. Hence, (3.3) along with (3.2c) give us the relations for calculation of L_n for scheme 2.

The first few values of L_n for scheme 1- 2 are given in Table 1., for different values of the capture capacity. As expected, scheme 2 outperforms scheme 1. This follows from the fact that in scheme 1, when $2 \leq n \leq \Gamma$, there is always a possibility that all the nodes in a capture set flip 1, which leads to a wasted idle slot. On the other hand, in scheme 2, for $2 \leq n \leq \Gamma$, there is no randomness in actions of the nodes in a capture set, i.e., all the nodes in the capture set transmit their packets following a capture slot, and therefore no slot will be wasted.

B. PCM - FWC

According to this model, when $n \geq 2$, with probability $P_c(n)$, one out of n contending packets is captured and correctly received.

Considering scheme 1, given that $n \geq 2$ users contend in the first slot of some CRI, with probability $P_c(n)$, we have the situation in Figure 3.1(a); while with probability $[1-P_c(n)]$, we have the situation in Figure 3.1(b). Hence, we obtain L_n as:

$$L_n = \frac{1 + 2 P_c(n) \sum_{i=0}^{n-1} L_i P_{n-1}(i) + 2 (1 - P_c(n)) \sum_{i=0}^{n-1} L_i P_n(i)}{1 - 2 (1 - P_c(n)) P_n(n)} \tag{3.5}$$

where again $L_0 = L_1 = 1$. Equation (3.5) is the desired recursion relation for the calculation of L_n.

Applying scheme 2 to this model; whenever a capture slot occurs, all the users in the capture set transmit their packets in the very next slot. Therefore, we obtain L_n as :

$$L_n = [1+P_c(n) L_{n-1} + 2 (1-P_c(n)) \sum_{i=0}^{n-1} L_i P_n(i)] / [1 - 2 (1-P_c(n)) P_n(n)] \tag{3.6}$$

Here again, for the same reason stated in III-A, scheme 2 outperforms scheme 1.

IV. CTMCRA FOR FEEDBACK WITHOUT CAPTURE

In feedback without capture (FWOC), the receiver cannot distinguish between capture and success slots. Here, the problem is to determine what action should be taken by the nodes from the capture set that transmit a packet and receive an ACK with a different node identity number. Only these nodes from the capture set know that the receiver has captured a packet, while other nodes consider the capture slot as a success one.

We consider two different courses of action, introduced in [5], that can be taken by the nodes from the capture set.

A. DCM - FWOC1 ; "Wait for Partial Conflict Resolution" Scheme

In this scheme, a CRI that corresponds to a given initial collision is divided into several parts. The first part ends when all nodes involved in the initial collision, except those nodes from capture sets that were involved in a capture slot in the first part, have successfully transmitted their packets. Those nodes from capture sets wait until the first part ends, and then retransmit their packets. These packets are called *residual packets*. The second part is dedicated to resolve conflicts (if any) among these nodes. Nodes from the capture sets of the second part (if any) will retransmit their packets in the third part and so on. The initial conflict is finally resolved when an empty part is detected (i.e., a single idle slot). An example of this scheme is given in Figure 4.1.

Denoting by L_n, the conditional mean of a CRI of conflict multiplicity n, and by $\tilde{L}_n$, the conditional mean of the first part of the same CRI; L_n can be written as

$$L_n = \tilde{L}_n + \sum_{m=0}^{n-1} Q_n(m)\, L_m \quad ; \quad n \geq 1, \tag{4.1}$$

with $L_0 = 1$. $Q_n(m)$ is defined as

$$Q_n(m) \equiv Pr(\text{exactly } m \text{ residual packets} \mid n \text{ packets in initial collision}).$$

The conditional mean of the first part of CRI is given by:

$$\tilde{L}_n = 1 \quad ; \quad n \leq \Gamma \tag{4.2a}$$

$$\tilde{L}_n = [1 + 2\sum_{i=0}^{n-1} \tilde{L}_i P_n(i)] \,/\, [1 - 2P_n(n)] \quad ; \quad n > \Gamma. \tag{4.2b}$$

Now, depending on the initial number of colliding packets, Γ, and results of actions taken by nodes in the first part, the number of residual packets which must be retransmitted in the second part can be between 0 and $n-1$. It is easy to see that $Q_0(0) = 1$; and for $0 < n \leq \Gamma$;

$$Q_n(m) = \begin{cases} 1 \ ; & m = n-1 \\ 0 \ ; & otherwise \end{cases} \tag{4.3a}$$

When $n > \Gamma$, there is a collision in the first slot of first part of CRI. With probability $P_n(i)$, i nodes flip 0 and retransmit in the next slot; that is a CRI of i nodes begins. With probability $Q_i(l)$, $0 \leq l \leq i$, the number of residual packets of this CRI is exactly l. Also, with probability $Q_{n-i}(m-l)$, the number of residual packets of CRI for remaining $n-i$ packets is $m-l$. Therefore the probability that the number of residual packets at the end of first part is exactly m is given by

$$Q_n(m) = \sum_{i=0}^{n} \sum_{l=0}^{\min(i,m)} P_n(i) Q_i(l) Q_{n-i}(m-l) \tag{4.3b}$$

It is easy to see that for $n \leq \Gamma$, $L_n = n+1$.

B. DCM - FWOC2 ; "Send in the Next Slot" Scheme.

In this scheme, whenever a node detects that it belongs to a capture set, it retransmit in the next slot. Here, a CRI consists of two parts. The first part evolves according to the rules of the CRA. The second part is dedicated to retransmission of packets that were involved in a capture event during the last slot of the first part. An example of this scheme is given in Figure 4.2.

Here again, $Q_n(m)$ represents the probability of exactly m residual packets, given that n packets were involved in the initial collision. When $n \leq \Gamma$, the initial conditions of $Q_n(m)$ are same as part A, when $n \leq \Gamma$.

For $n > \Gamma$, there is a collision in the first part of the CRI. With probability $P_n(i)$, i nodes flip "zero" and retransmit in the next slot; that is a CRI of i nodes begins. With probability $Q_i(l)$, $0 \le l \le i$, the number of residual packets of this CRI is exactly l. Therefore, with probability $Q_i(l)$, the next CRI begins with $n-i+l$ nodes. Hence, with probability $Q_{n-i+l}(m)$, we have m residual packets at the end of the first part of CRI.

It is important to observe that the number of residual packets is less than or equal to $\Gamma-1$. This is due to the fact that the residual packets are those packets that were involved in a capture slot in the last time slot of preceding CRI part.

From the above observation, we can obtain a recursian formula for residual probabilities. This can be written as

$$Q_n(m) = \sum_{i=0}^{n} \sum_{l=0}^{i} P_n(i) Q_i(l) Q_{n-i+l}(m) \quad ; \quad 0 \le m \le \Gamma-1, \tag{4.4a}$$

and $Q_n(m)=0$ for $m \ge \Gamma$.

It is easy to see that for $i < j$, $Q_n(i)$ depends on $Q_n(j)$, and therefore, for calculation of these probabilities, we start at $m=\Gamma-1$ and move down to $m=0$.

Once we obtained the probabilities of residual packets, we can proceed to calculation of conditional mean of CRI length. It is easy to see that (4.1) still holds, although $\tilde{L}_n$ has a different formulation. Since $L_n = n+1$ for $n \le \Gamma$, and $Q_n(m)$, $0 \le m \le \Gamma-1$, is the probability mass function of the number of residual packets, we have

$$L_n = 1 + \tilde{L}_n + \sum_{m=1}^{\Gamma-1} m \; Q_n(m) \quad ; \quad n \ge 1. \tag{4.5}$$

Furthermore, $\tilde{L}_n = 1$ for $n \le \Gamma$; and for $n > \Gamma$:

$$\tilde{L}_n = \sum_{i=0}^{n} [1 + \tilde{L}_i + \sum_{m=0}^{i} Q_i(m) \tilde{L}_{n-i+m}] P_n(i) \quad ; \quad n > \Gamma. \tag{4.6a}$$

Simplifying the above, we obtain the recursion relation for $\tilde{L}_n$:

$$\tilde{L}_n = [1 + \sum_{i=1}^{n-1} P_n(i) [\tilde{L}_i + \sum_{m=0}^{\min(i,\Gamma-1)} Q_i(m) \tilde{L}_{n-i+m}] + 2\, P_n(n)] \; / \; [1 - 2\, P_n(n)] \tag{4.6b}$$

C. PCM - FWOC1 ; "Wait for Partial Conflict Resolution" Scheme

In PCM, with probability $P_c(n)$, one out of n contending packets is captured. Following the rules of this scheme, we observe that for the first part of the CRI, we have:

$$\tilde{L}_n = P_c(n) + (1 - P_c(n)) \sum_{i=0}^{n} (1 + \tilde{L}_i + \tilde{L}_{n-i}) P_n(i) \tag{4.7}$$

where $\tilde{L}_0 = \tilde{L}_1 = 1$.

To calculate the probability of residual packets from the first part, we know the following initial conditions:

$$Q_0(0) = Q_1(0) = 1 \tag{4.8a}$$

and given that a capture has occurred

$$Q_n(m \mid c) = \begin{cases} 1 \; ; & m = n-1 \\ 0 \; ; & otherwise \end{cases} \tag{4.8b}$$

Therefore, for $n \geq 2$ and $0 \leq m \leq n-1$:

$$Q_n(m) = P(n)\,\delta_{m,n-1} + (1 - P_c(n)) \sum_{i=0}^{n} \sum_{l=0}^{\min(i,m)} P_n(i) Q_i(l) Q_{n-i}(m-l) \tag{4.8c}$$

where $\delta_{i,j}$ is the Kronecker delta defined to be 1 if $i = j$, and 0 if $i \neq j$. Finally, the conditional mean of the CRI length is obtained using (4.1).

V. MAXIMAL STABLE THROUGHPUT FOR POISSON PACKET ARRIVALS

Let the packet generation process be a stationary Poisson point process with parameter λ. Denoting by Y_i, the length of the i-th CRI, it can be easily shown that $\{Y_i\}$ is a Markov chain. The condition for the ergodicity of this chain leads to obtaining a linear upper bound on L_n ; i.e.,

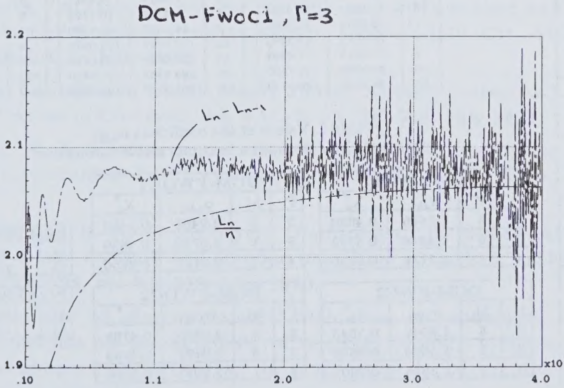
$$L_n \leq \alpha_{uM} . n + \beta \quad ; \qquad n > M \tag{5.1}$$

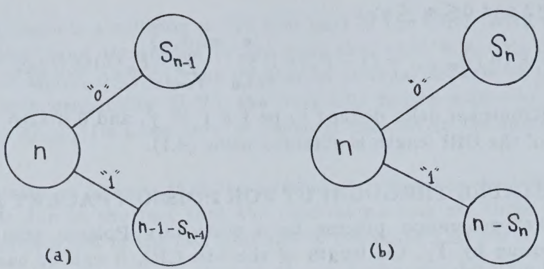
where α_{uM} and β are constants depending on the parameters of RAS (e.g., capture parameters, bias of binary coins, etc.). The stability of CRA is then gauranteed whenever $\lambda < 1 / \alpha_{uM} \equiv \lambda_u^*$. This means that the asymptotic values of CRI length and number of packets waiting for (re)transmission remains finite as long as the new packet process rate, λ, is less than λ_u^*.

The analytical derivation of α_{uM} seems to be intracable, so once we evaluate L_n, we use numerical computation to find the upper bound on the maximal stable throughput of system.

Table 2. gives the maximal stable throughput for DCM and the two different models of feedback. In the case of FWC, scheme 2 always performs better than scheme 1; but, in case of FWOC, for smaller values of contending nodes, scheme 1 performs better than scheme 2, while scheme 2 outperforms scheme 1 for larger number of contending nodes. This behavior can be observed from the inspection of Table 3. For example, for $\Gamma = 4$, scheme 1 is a better option up to 329 contending nodes. As can be seen from Table 3., as we increase the capture capacity, Γ, scheme 1 performs better for larger number of contending users. This characteristic of "send in the next slot" scheme arises from the fact that for larger values of capture capacity, after each capture slot, it is more likely to have a collision slot which in turn prolongs the CRI.

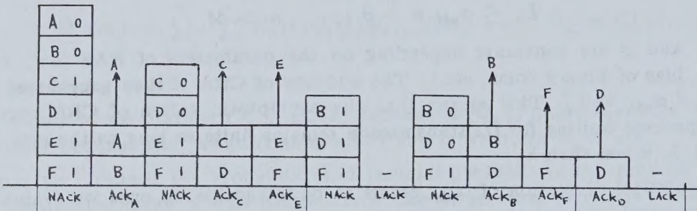
It is shown that L_n is not exactly a linear function of n and has an oscillatory behavior. It is observed that as capture capacity of RAS increases, this oscillatory behavior becomes more prominent. Although we have not addressed this problem rigorously, it is observed that this behavior effects the bounding of α_{uM} and hence λ_u^*.





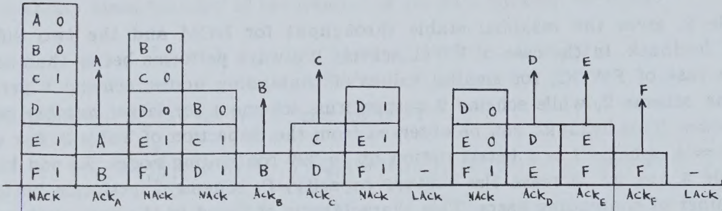
S_{n-1}
"0"
n
"1"
$n-1-S_{n-1}$
(a)
S_n
"0"
n
"1"
$n-S_n$
(b)

Fig. 3.1: General action of the CTMCRA in first slot of CRI,
(a) $2 \leq n \leq \Gamma$, (b) $n > \Gamma \geq 2$.



A 0
B 0
C I
D I
E I
F I
NAck
A
A
B
Ack_A
C 0
D 0
E I
F I
NAck
C
C
D
Ack_C
E
E
F
Ack_E
B I
D I
F I
NAck
–
LAck
B 0
D 0
F I
NAck
B
B
D
Ack_B
F
F
Ack_F
D
D
Ack_D
–
LAck

Fig. 4.1: Example of DCM-FWOC1 ; $\Gamma = 2$.



A 0
B 0
C I
D I
E I
F I
NAck
A
A
B
Ack_A
B 0
C 0
D 0
E I
F I
NAck
B 0
C I
D I
NAck
B
B
Ack_B
C
C
D
Ack_C
D I
E I
F I
NAck
–
LAck
D 0
E 0
F I
NAck
D
D
E
Ack_D
E
E
F
Ack_E
F
F
Ack_F
–
LAck

Fig. 4.2: Example of DCM-FWOC2 ; $\Gamma = 2$.

Table 1. Expected CRI length for different values of capture capacity, (DCM-FWC ; $L_n^i \equiv L_n$ for scheme i)

$\Gamma = 2$		
n	L_n^1	L_n^2
2	3.0000	2.0000
3	5.6667	4.6667
4	7.6667	6.2381
5	9.8000	7.9905
6	11.9720	9.8015
7	14.1472	11.6203
8	·16.3165	13.4321
9	18.4808	15.2367
10	20.6427	17.0375
$\Gamma = 3$		
n	L_n^1	L_n^2
2	3.0000	2.0000
3	4.0000	3.0000
4	6.7143	5.2857
5	8.3714	6.5619
6	10.1595	7.9889
7	12.0145	9.4877
8	13.8928	11.0083
9	15.7716	12.5275
10	17.6433	14.0381
$\Gamma = 4$		
n	L_n^1	L_n^2
2	3.0000	2.0000
3	4.0000	3.0000
4	5.2500	4.0000
5	7.8833	6.1333
6	9.3565	7.2839
7	10.9491	8.5522
8	12.6263	9.8962
9	14.3472	11.2768
10	16.0812	12.6665

Table 3. Expected CRI length for different values of capture capacity, (DCM-FWOC ; $L_n^i \equiv L_n$ for scheme i)

$\Gamma = 2$			$\Gamma = 3$			$\Gamma = 4$		
n	L_n^1	L_n^2	n	L_n^1	L_n^2	n	L_n^1	L_n^2
1	2.0000	2.0000	1	2.0000	2.0000	1	2.0000	2.0000
2	3.0000	3.0000	2	3.0000	3.0000	2	3.0000	3.0000
3	5.6667	5.2738	3	4.0000	4.0000	3	4.0000	4.0000
4	7.2381	7.3952	4	6.2857	6.1075	4	5.0000	5.0000
5	8.9905	9.4421	5	7.5619	8.1577	5	7.1333	7.0500
6	11.1472	11.4841	6	9.4037	10.1780	6	8.2839	9.0742
7	13.5052	13.5391	7	11.5337	12.1900	7	10.1818	11.0848
8	15.8010	15.5992	62	123.0227	123.0573	8	11.9910	13.0907
9	17.9618	17.6584	63	125.0878	125.0729	329	656.1511	656.1694
10	20.0362	19.7162	99	199.8441	197.6436	330	658.2386	658.2357
59	128.8335	120.5153	100	201.9127	199.6602	340	678.3911	678.1696

Table 2. Values of the coefficient α_{uM} ;
upper and lower bounds on maximal stable throughput.

DCM-FWC1			
Γ	M	α_{uM}	λ_u^*
2	5	2.1646	0.4620
3	6	1.8648	0.5363
4	7	1.7158	0.5830

DCM-FWOC1			
Γ	M	α_{uM}	λ_u^*
2	5	2.2468	0.4450
3	6	2.0750	0.4820
4	7	2.0197	0.4950

PCM-FWC1 (DS/SS, $N = 31$)		
M	α_{uM}	λ_u^*
13	1.5185	0.6585

DCM-FWC2			
Γ	M	α_{uM}	λ_u^*
2	5	1.8042	0.5543
3	6	1.5064	0.6638
4	7	1.3695	0.7301

DCM-FWOC2			
Γ	M	α_{uM}	λ_u^*
2	5	2.0885	0.4788
3	6	2.0162	0.4959
4	7	1.9985	0.5004

PCM-FWC2 (DS/SS, $N = 31$)		
M	α_{uM}	λ_u^*
13	1.2383	0.8076

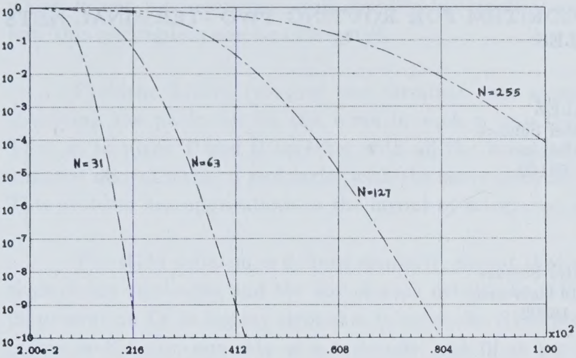


Figure c1. Probability of Capture vs. number of contending nodes for BPSK DS/SSMA ($\frac{E_b}{N_o} = 16$ dB, $L = 1000$ bits/packet, $Q = 0.01$, $m_\phi = \frac{1}{3}$)

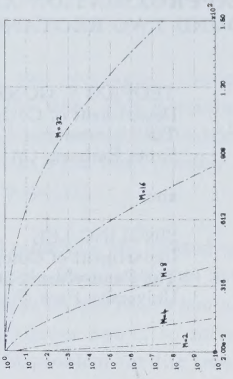


Figure c2 Probability of Capture vs. number of contending nodes for MFSK SFH/SSMA ($\frac{E_b}{N_o} = 16$ dB, $L = \frac{1000}{\log_2 M}$, $Q = 0.01$, $q = 100$, $N_s = 10$)

REFERENCES

[1] J. I. Capetanakis, "The multiple access broadcast channel: protocol and capacity considerations," Ph.D. Thesis, course VI, Mass. Inst. Tech., Cambridge, MA, August 1977.

[2] B. S. Tsybakov and V. A. Mikhailov, "Free synchronous packet access in a broadcast channel with feedback," Probl. Peredachi Inform., vol. 14, pp.32-59, Oct.-Dec. 1978.

[3] J. L. Massey, "Collision-resolution algorithms and random access communications," in G. Longo, Ed. Multi-user Communications. New York: Springer-Verlag, CISM Courses and Lecture Series, no. 265, 1981.

[4] R. G. Gallager, "Conflict resolution in random access broadcast networks," in Proc. AFOSR Workshop Commun. Theory and Appl., Provincetown, MA, Sept. 1978, pp. 74-76, 17-20.

[5] I. Cidon and M. Sidi, "The effect of capture on collision resolution algorithms," IEEE Trans. Commun., vol. COM-33, pp. 317-324, April 1985.

[6] I. Cidon and M. Sidi, "Splitting protocols in presense of capture," IEEE Trans. Info. Th. vol. IT-31, pp. 295-301, March 1985.

[7] L. G. Roberts, "ALOHA packet system with and without slots and capture," ASS Note 8, Advanced Research Projects Agency, Network Info. Ctr., Stanford Research Inst., Stanford, CA, June 1972.

[8] C. Namislo, "Analysis of Mobile Radio Slotted ALOHA Networks," IEEE Journal on Selected Areas in Commun. vol. SAC-2, pp. 583-588, July 1984.

[9] R. E. Kahn, S. A. Gronemeyer, J. Burcfiel, and R. C. Kunzelman, "Advances in packet radio technology," Proceedings of the IEEE, 66, pp. 1468-1496, November 1978.

[10] D. H. Davis and S. A. Gronemeyer, "Performance of slotted ALOHA Random Access with Delay Capture and Randomized Time of Arrival," IEEE Trans. Commun., vol. COM-28, pp. 703-710, May 1980.

[11] E. Geraniotis and M. Soroushnejad, "Evaluation of probability of capture and study of thresholds on the capability of rejecting primary multiple-access interference in spread-spectrum networks," in preparation to be submitted to the IEEE Trans. on communications.

AN APPROXIMATION ALGORITHM FOR ROUTING TWO -TERMINAL NETS AROUND TWO RECTANGLES

TEOFILO F. GONZALEZ
Department of Computer Science
The University of California
Santa Barbara, CA 93106

and

SINGLING LEE
Department of Computer Science
The Pennsylvania State University
University Park, PA 16802

ABSTRACT

The problem of connecting a set of 2m terminals that lie on the sides of two equal-width rectangles to minimize the total area is discussed. We present an O(m log m) approximation algorithm to solve this problem. Our algorithm generates a solution with area $\leq 2 * \text{OPT}$, where OPT is the area of an optimal solution. The nets are routed according to the following greedy strategy: the wire connecting the two points in a net is one whose path crosses the least number of corners of the rectangles. For some nets there is more than one path that crosses the least number of corners of two rectangles. These nets are connected by wires whose paths blend with the paths for other nets.

I. INTRODUCTION.

Let T (top) and B (bottom) be two rectangles with equal width (possibly different heights) and placed on the same plane with the same orientation. The left side of T and B have been placed along the same vertical line and rectangle T is above retangle B. The distance between these two rectangles is not fixed and will be decided by our routing algorithm. This distance must be at least λ units. Let S be a set of 2m terminals that lie on the sides of T and B, and let N_1, N_2, ..., N_m be any partition of set S such that the size of each subset is exactly two. Each subset N_i is called a net. All the points in each net have to be made electrically common by interconnecting them with wires. The wires follow a path consisting of a finite number of horizontal and vertical line segments. These segments are assigned to two different layers. All the horizontal segments are assigned to one layer and all the vertical segments are assigned to the other layer. Line segments on different layers can be connected directly at any given point z by a wire perpendicular to the layers if both line segments cross point z in their respective layers. Every pair of distinct and parallel line segments must be at least λ units apart and every line segment must be at least λ units from each side of rectangles T and B, except in the region where the path joins a point in S to connect it. Also no path is allowed inside of T and B on any of the layers. We are assuming that no two terminals in the middle channel (the region between the bottom side of rectangle T and the top side of rectangle B) are placed along the same vertical line. This assumption, which is not too restrictive, will simplify our routing algorithm. Later on we explain how this assumption can be eliminated.

Problem 2-R2M (routing two terminal nets around two rectangles) consists of specifying the paths for all the wires in such a way that the total area is minimized. That is, to place T and B together with all the wires (that must satisfy the restrictions imposed above) inside a rectangle (with the same orientation as T) of least possible area. This problem has applications in the layout of integrated circuits [11].

The R1M problem is defined similarly, except that all terminals are located on the sides of one rectangles and the size of each net can be arbitrary. Hashimoto and Stevens [8] present an O(m log m) algorithm to solve the R1M problem for the case when all the points in S lie on one side of a rectangle. An Ω(m log m) lower bound to solve this problem was established in [7]. Algorithms to solve the R1M problem for the case that all nets are restricted to be of size two (called 2-R1M problem) appear in [9] and [:]. The one in [3] is optimal with respect to the time complexity bound. Several approximation algorithms for the R1M problem have been given by Gonzalez and Lee ([4] , [5] and [6]). The time complexity of all these algorithms is O (m*(n + log m)) and the best of these algorithms generates a solution with area $\leq$ 1.6 * OPT, where OPT is the area of an optimal solution, m is the number of terminals and n is the number of nets. If more than two layers are allowed and wire overlap is permitted, the R1M problem becomes an NP-hard problem [14], even when the size of all nets is two. Baker [1] presents an O(m log m) algorithm for the 2-R2M. This algorithm generates a solution within 1.9 of optimal. For this case the optimality is measured with respect to the perimeter of the smallest enclosing rectangle. In this paper we present an approximation algorithm for the 2-R2M problem that generates solutions within a factor of 2 of the optimal solution. Our algorithm is simpler than the one appears in [1]. Our lower bound is obtained by similar technique. However, our analysis is much simpler than the one appears in [1]. One of the reason is our assumption for the terminals located in the middle channel.

Let 2_S-*R2M* (2_G-*R2M)* denote a 2-R2M problem in which for each global net there exists exactly one (more than one) connecting path that crosses the least number of corners. In Section II we introduce our notation and present some basic results. In order to simplify the exposition of our results, we begin by presenting approximation algorithms for restricted versions of the 2-R2M problem. In section III we present an approximation algorithm for the 2_S-R2M problem. An approximation algorithm for the 2_G-R2M is presented in the section IV and in section V we indicate how to combine these results to obtain our approximation algorithm.

II. NOTATION AND BASIC RESULTS.

We begin by defining the 2-R2M problem and introducing notation similar to the one in [2]. Let T and B be two rectangular components located on the same plane with size h_T by w_T (height by width) and h_B by w_B, respectively. We assume that $w_T = w_B$. Rectangle T is above rectangle B and the distance between these two rectangles is not fixed, but must be at least λ units. The left side of T and B have been placed along the same vertical line. Since T and B have the same width, the right side of T and the right side of B are also located along the same vertical line. There are 2m terminals (T_1 , T_2 , ... , T_{2m}) on the sides of T and B. The set of terminals is partitioned into m subsets denoted by N_1 , N_2 , ... , N_m and each subset contains exactly two terminals. Each subset N_i is called a *net*. It is assumed that every pair of terminals is at least λ units apart and every terminal is located at least λ units from each of the corners of T and B. All the terminals in each net must be made electrically common by connecting them with wires. The path followed by these wires consists of a finite number of horizontal and vertical line segments. Each of these line segments must lie on the same plane as T and B, be parallel to a side of T and B, and be on the outside of T and B. Perpendicular line segments can intersect at any point, but parallel line segments must be at least λ units apart. Also, all line segments must be at least λ units away from every side of rectangles T and B except in the vicinity where a line segment connects a terminal. The 2-R2M problem consists of specifying paths for all the interconnections subject to the rule mentioned above in such a way that the total area is minimized, i.e., place the components T and B together with all the wires inside a rectangle (with the same orientation as T and B) of least possible area.

Label the sides of the components (in the obvious way) left, top, right and bottom. Starting in the bottom-left corner of T, traverse the sides of rectangle T clockwise. The ith corner visited is labeled S_{i-1} and the ith terminal visited is terminal T_i. Assume that terminal T_{m_1} is the last terminal visited by this procedure. Using a similar procedure traverse the sides of rectangle B, the ith corner of B visited is labeled R_{i-1} and the ith terminal (that lies on the sides of B) visited is terminal T_{m_1+i}. The *close interval [x ,y]* consists of all the points on the sides of rectangle X ,Y $\in$ {T , B} that are visited while traversing the sides of rectangle X in the clockwise direction starting at point x and ending at point y. Note that the interval [x , x] consists of a single point. Parentheses are used instead of brackets for open intervals. We use [S_0 , S_1], [S_1 , S_2], [S_2 , S_3] and [S_3 , S_0] ([R_0 , R_1], [R_1 , R_2], [R_2 , R_3] and [R_3 , R_0]); T^l, T^t, T^r and T^b; (B^l, B^t, B^r and B^b) to represent the left, top, right and bottom sides of T (B), respectively. The function C(j) indicates the index of the terminal to which terminal T_j has to be connected.

In order to simplify our notation we introduce the following lemma which can be proved by simple interchange arguments.

Lemma 2.1: Let W be any layout for an instance of the 2-R2M problem. Let T_q be any terminal located on the bottom (top) side of T (B). If T_q and $T_{C(q)}$ are located on different rectangles and T_q is connected to $T_{C(q)}$ by a path that crosses the top (bottom) side of rectangle T (B), then there exists another layout W' such that T_q is connected to $T_{C(q)}$ by a path that does not cross top (bottom) side of rectangle T (B), and the area of W' is not larger than the area of W.
Proof: For brevity the proof is omitted. □

By lemma 2.1 whenever we refer to an optimal layout, we may assume it cannot be transformed by applying the interchange argument given in lemma 2.1. In the layout that our algorithm generates we cannot apply the transformation implied by lemma 2.1. Note that this assumption will not make the problem easier, it will just allow us to use a simpler notation.

Set $D = \{ (d_1,x(a_1),y(a_1)), (d_2,x(a_2),y(a_2)), \ldots , (d_m,x(a_m),y(a_m)) \}$, where $1 \leq a_i \leq 2m$ and the functions x and y are defined from $\{1,2,\ldots,2m\}$ to $\{C,R\}$, is said to be an *assignment* if $|\{ a_1,C(a_1), a_2,C(a_2), \ldots , a_m,C(a_m) \}| = 2m$. Any subset of an assignment is said to be a *partial assignment*. An assignment D indicates the direction of the path connecting the two points in each net. If $(i,x(i),y(i)) \in D$, then the path connecting this net is as follows. If T_i and $T_{C(i)}$ are located on the same rectangle then $x(i) \neq y(i)$ and the path connecting this nets starts at terminal T_i ($T_{C(i)}$) if $x(i) = C$ ($x(i) \neq C$) moving perpendicular to the side that contains this terminal and then it continues in the clockwise direction (with respect to the rectangle) until it reaches point $T_{C(i)}$ (T_i). This second terminal is joined to this path by a line segment perpendicular to the side where it is located. On the other hand if T_i and $T_{C(i)}$ are located on different rectangles, assume that T_i is located on rectangle T, the path connecting this net consists of three segments. The first segment starts at terminal T_i moving perpendicular to the side that contains this terminal and then it continues in the clockwise direction, if $x(i) = C$ otherwise counterclockwise direction, until it reaches either the point to be connected , or the bottom-left or bottom-right corner of rectangle T. The second segment starts at terminal $T_{C(i)}$ moving perpendicular to the side which contains this terminal and then it continues in the clockwise direction, if $y(i) = C$ otherwise counterclockwise direction, until it reaches either the the point to be connected, or the top-left or top-right corner of rectangle B. The third segment will join the end points of the previous two segments. Note that this segment could be a horizontal line segment through the middle channel. Note that a path that connects a net which could be changed by applying the transformation implied by lemma 2.1 cannot be represented by an assignment. Also, it is simple to see that doglegs can not be represented by an assignment.

For any $(k, x(k) , y(k)) \in D$, we say that the path connecting T_k , $T_{C(k)}$ given by D *crosses point z* (z is on the sides of T and B but not a corner point) if the connecting path intersects a line segment perpendicular to the side where z is located that starts at point z and ends at the point where it intersects the rectangle where z is not located (note that if it never intersects the rectangle, it is a half line). Introduce all horizontal and vertical half lines that intersect exactly one corner point. Introduce all vertical line segments that intersect exactly one corner of each of the rectangles. It is simple to verify that the above procedures introduce exactly two line segments (one vertical and one horizontal) that intersect each of the corner points of the rectangles. We called these segments extension lines. A connecting path *vertically (horizontally) crosses corner z* if the connecting path crosses the horizontal (vertical) extension associated with corner z.

For any assignment (or partial assignment) D we define the *height function* H_D for $[x , y]$ representing a side in one of the rectangles. as follows:

$$H_D(x , y) = \max\{ \text{number of the paths given by D that cross point } z \mid z \in (x,y)\}$$

Similarly, for any assignment D we define height function VE_D and HO_D for corner point z:

$$ve_D(z) = \text{number of the paths given by D that vertically cross point z.}$$

$$ho_D(z) = \text{number of the paths given by D that horizontally cross point z.}$$

The *height of assignment D for side X of rectangle T (B) (X ∈ {top , left , right, bottom})* refers to the value of $H_D(x , y)$, where $[x , y]$ is the interval that represents side X of rectangle T (B). For an assignment D, we define $H_D(R_0 , S_1)$ and $H_D(S_2 , R_3)$ as follows:

$$H_D(R_0 , S_1) = \max(H_D(S_0 , S_1) , H_D(R_0 , R_1)), \text{ and}$$
$$H_D(S_2 , R_3) = \max(H_D(S_2 , S_3) , H_D(R_2 , R_3)).$$

The next two lemmas establish that the 2-R2M problem reduces to the problem of finding an assignment D with least

$$(h_T+h_B + (H_D(S_1 , S_2) + H_D(R_1 , R_2) + H_D(R_3 , R_0)) * \lambda) * (w_T+ H_D(R_0 , S_1) + H_D(S_2 , R_3)) * \lambda)$$

and then in O(n log n) time one can construct a layout of area h_Q by w_Q for it.

Lemma 2.2: For every assignment D, there is a rectangle Q of size h_Q by w_Q, where

$$h_Q = h_T + h_B + (H_D(S_1 , S_2) + H_D(R_1 , R_2) + H_D(R_3 , R_0)) * \lambda, \text{ and}$$
$$w_Q = w_T+(H_D(R_0 , S_1) + H_D(S_2 , R_3)) * \lambda.$$

with the property that rectangle T and B together with the interconnecting paths defined by assignment D can be made to fit inside Q.
Proof: The proof is a direct generalization of the proof for the R1M problem that appears in [10]. One additional vertical track might be required when there is no nets with exactly one terminal located in the middle channel [13]. □

Lemma 2.3: For any assignment D a layout with the area given by lemma 2.2 can be obtained in O(n log n) time.
Proof: The proof of this lemma is a straight forward generalization of the proof for the R1M problem that appears in [10]. The algorithm that constructs the final layout uses as a subalgorithm the procedure given in [7], [8] and [13]. □

Net N_i is said to be a *local net* if its two terminals are located on the same side of one rectangle or if one terminal is located on the bottom side of T and the other is located on the top side of B. Otherwise, net N_i is said to be *global.* Let $X^j Y^k$ represent the set of nets in which one of its terminals is located on side j of rectangle X and the other is located on side k of rectangle Y, where X,Y ∈ {T , B} and j , k ∈ {l , t, r, b}. Let $|X^j Y^k|$ represent the number of nets in this set. For an assignment D we define the *function A(D)* as

$$(h_T+h_B + (H_D(S_1, S_2) + H_D(R_1, R_2) + H_D(R_3, R_0)) * \lambda) * (w_T + H_D(R_0, S_1) + H_D(S_2, R_3)) * \lambda).$$

i.e., the total area required for a layout of T and B and all interconnections given by D.

Definition 2.1: D'

Let D' be the partial assignment in which all the local nets are connected by paths that do not cross any of the corners of T and B.

Lemma 2.4: Every assignment D can be transformed to an assignment M such that $D' \subseteq M$ and $A(M) \leq A(D)$.
Proof: The proof follows the same lines as the one for the 2-R1M problem that appears in [10]. □

It should be clear that it is only required to specify the paths connecting all the global nets since we know that the local nets can be routed optimally by routing them as indicated in assignment D'. Also, once we have an assignment the proof of lemma 2.3 (a constructive proof) can be used to find a layout of optimal area for it.

III. APPROXIMATION ALGORITHM FOR THE 2_S-R2M PROBLEM.

In this section we present our approximation algorithm for the 2_S-R2M problem. Let M_S be the set of global nets for which the number of distinct connecting paths that cross the least number of corners is exactly one. Clearly,

$$M_S = \{ T^fT^g, B^fB^g \mid f,g \text{ are adjacent sides on the same rectangle}\} \\ \cup \{T^jB^k, T^kB^j \mid j \in \{l, t, b, r\} \text{and } k \in \{l, r\} \}.$$

For the set of nets M_S, we construct assignment D_S as follows:

$$D_S = \{ \text{all nets in } M_S \text{ are connected by paths that cross the least number of corners of T and B}\}.$$

Lemma 3.1: Let D be an optimal assignment such that $D' \subseteq D$. Let M be D except that all nets in T^fT^g and B^fB^g, where f and g are adjacent sides on the same rectangles, are assigned as in our algorithm. Then

$$H_M(S_1,S_2)+H_M(R_1,R_2)+H_M(R_3,R_0) \leq H_D(S_1,S_2)+H_D(R_1,R_2)+H_D(R_3,R_0), \text{ and}$$

$$H_M(R_0,S_1)+H_M(S_2,R_3) \leq H_D(R_0,S_1)+H_D(S_2,R_3)+ \max(\sum_{\substack{f,g\\ \text{adjacent}}} | Y^fY^g |, \sum_{\substack{f,g\\ \text{adjacent}}} (| Z^fZ^g |).$$

where Y^fY^g (Z^fZ^g) is the subset of nets in T^fT^g (B^fB^g) that are connected differently in D and M.
Proof: For brevity the proof is omitted. □

set	contribution to our lower bound for $w_T/(\lambda)$ + $H_D(R_0,S_1)+H_D(S_2,R_3)$	contribution to our lower bound for $(h_T+h_B)/(\lambda)$ + $H_D(S_1,S_2)+H_D(R_1,R_2)+H_D(R_3,R_0)$	
T^fT^f	$0.5*(\lvert T^fT^g\rvert+\lvert Y^fY^g\rvert)$	$\lvert T^fT^g\rvert+\lvert Y^fY^g\rvert$	f,g adjacent
B^fB^f	$0.5*(\lvert B^fB^g\rvert+\lvert Z^fZ^g\rvert)$	$\lvert B^fB^g\rvert+\lvert Z^fZ^g\rvert$	f,g adjacent
T^jB^k	$0.5*(\lvert T^jB^k\rvert+\lvert Y^jZ^k\rvert)$	$2*\lvert T^jB^k\rvert+\lvert Y^jZ^k\rvert$	$j,k \in\{l,r\}$
T^tB^k	$\lvert T^tB^k\rvert$	$2*\lvert T^tB^k\rvert$	$k\in\{l,r\}$
T^bB^k	$0.5*(\lvert T^bB^k\rvert+\lvert Y^bZ^k\rvert)$	$\lvert T^tB^k\rvert+\lvert Y^bZ^k\rvert$	$k\in\{l,r\}$
T^kB^t	$0.5*(\lvert T^kB^t\rvert+\lvert Y^kZ^t\rvert)$	$\lvert T^kB^t\rvert+\lvert Y^kZ^t\rvert$	$k\in\{l,r\}$
T^kB^b	$\lvert T^kB^b\rvert$	$2*\lvert T^kB^b\rvert$	$k\in\{l,r\}$

Table 3.1: Lower bounds for the 2_S-R2M problem.

Lemma 3.2: Let D be an optimal assignment such that D' $\subseteq$ D. Let M be D except that all nets in $T^jB^k \cup T^kB^j$, where $j \in \{l, t, b, r\}$, $k \in \{l, r\}$ are assigned as in our algorithm. Then

$$H_M(S_1,S_2)+H_M(R_1,R_2)+H_M(R_3,R_0) \leq H_D(S_1,S_2)+H_D(R_1,R_2)+H_D(R_3,R_0)\text{ , and}$$

$$H_M(R_0,S_1)+H_M(S_2,R_3) \leq H_D(R_0,S_1)+H_D(S_2,R_3)+ \sum_{\substack{j\in\{l,t,b,r\}\\ k\in\{l,r\}}} \lvert Y^jZ^k\rvert + \sum_{\substack{j\in\{t,b\}\\ k\in\{l,r\}}} \lvert Y^kZ^j\rvert .$$

where Y^jZ^k (Y^kZ^j) is the subset of T^jB^k (T^kB^j) that are connected differently in D and M.
Proof: For brevity the proof is omitted. □

Before proving our main result in this section we establish a lower bound on the area required by an optimal solution. Note that from lemma 3.1 and 3.2 we know that our algorithm generates a solution with minimum height for the 2_S-R2M problem. The lower bounds are given in table 3.1.

Lemma 3.3: Let D be an optimal assignment such that D' $\subseteq$ D. Assignment D and rectangles T and B satisfy the lower bound given in table 3.1, where Y^fY^g (Z^fZ^g) is the subset of nets in T^fT^g (B^fB^g) and Y^jZ^k (Y^kZ^j) is the subset of nets in T^jB^k (T^kB^j), that are connected differently in D and M.
Proof: For brevity the proof is omitted. □

Theorem 3.1: For the 2_S-R2M problem, let D be an optimal assignment such that D' $\subseteq$ D and let M be the assignment generated by our algorithm. Then $A(M) \leq 2 * A(D)$.
Proof: For brevity the proof is omitted. □

IV. APPROXIMATION ALGORITHM FOR THE 2_G-R2M PROBLEM.

In this section we present an approximation algorithm for the 2_G-R2M problem. Let M_G be the set of global nets for which the number of distinct connecting paths that cross the least number of corners is more than one. Clearly,

$$M_G = \{\, T^fT^g ,\ B^fB^g \mid f, g \text{ are two opposite sides of one rectangle}\} \cup \{T^tB^k ,\ T^kB^b \mid k \in \{t , b\} \}.$$

In what follows we only explain how the assignment $D_G(T^tB^b)$ for all nets in T^tB^b is constructed, since all assignments $D_G(X^jY^k)$, where $X^jY^k \in \{ T^lT^r, B^lB^r, T^tT^b, B^tB^b, T^tB^t, T^bB^b, T^tB^b \}$, are constructed by similar procedures. For any permutation, π, of the nets in set T^tB^b we define an assignment ASG(T^tB^b , π) as follows: the first net in π is connected by a path that begins on side T^t , crosses the left side of rectangle T and B and ends at side B^b; and the path connecting the kth net ($1 \leq k \leq |T^tB^b|$) in π begins on side B^b (T^t), crosses the right (left) side of T and B and ends at side T^t (B^b), when k is even (odd). We claim that there is a permutation, π, of the nets in set T^tB^b such that there is a layout for assignment ASG(T^tB^b , π) with the property that for any k ($1 \leq k \leq |T^tB^b|$) the path connecting the kth net in π can share the same track on the side where the path connecting the (k-1)st net ends. In this case we say that π is a *valid permutation* for the set of nets T^tB^b.

Claim: There is a valid permutation for the set of nets X^jY^k, where $X^jY^k \in \{ T^lT^r, B^lB^r, T^tT^b, B^tB^b, T^tB^t, T^bB^b, T^tB^b \}$.
Proof: For brevity the proof is omitted. □

set	contribution to our lower bound for $w_T/(\lambda)$ + $H_D(R_0,S_1)+H_D(S_2,R_3)$	contribution to our lower bound for $(h_T+h_B)/(\lambda)$ + $H_D(S_1,S_2)+H_D(R_1,R_2)+H_D(R_3,R_0)$
$T^lT^r \cup B^lB^r$	$0.5*(\lvert T^lT^r\rvert+\lvert B^lB^r\rvert)$	$2 *(\lvert T^lT^r\rvert+\lvert B^lB^r\rvert)$
$T^tT^b \cup B^tB^b$	$(\lvert T^tT^b\rvert+\lvert B^tB^b\rvert)$	$(\lvert T^tT^b\rvert+\lvert B^tB^b\rvert)$
$T^tB^t \cup T^bB^b$	$(\lvert T^tT^b\rvert+\lvert B^tB^b\rvert)$	$(\lvert T^tT^t\rvert+\lvert B^bB^b\rvert)$
T^tB^b	$1.5 *\lvert T^tB^b\rvert$	$\lvert T^tB^b\rvert$

Table 4.1: Lower bounds for the 2_G-R2M problem.

Lemma 4.1: Let D be an optimal assignment such that D' ⊆ D. Let M be D except that all nets in X^jY^k, where $X^jY^k \in \{ T^lT^r, B^lB^r, T^tT^b, B^tB^b, T^tB^t, T^bB^b, T^tB^b \}$, are assigned as in our algorithm. Then

$$H_M(S_1,S_2)+H_M(R_1,R_2)+H_M(R_3,R_0) \leq H_D(S_1,S_2)+H_D(R_1,R_2)+H_D(R_3,R_0) + |X^jY^k|,$$

where $X^jY^k \in \{ T^tT^b, B^tB^b, T^tB^t, T^bB^b, T^tB^b \}$,

$$H_M(S_0,S_1)+H_M(S_2,S_3) \leq H_D(S_0,S_1)+H_D(S_2,S_3)+ x *|X^jY^k|,$$

where $x = 1$ if $X^jY^k = T^lT^r$, $x = 0.5$ if $X^jY^k \in \{ T^tT^b, T^tB^b, T^tB^t \}$, and

$$H_M(R_0,R_1)+H_M(R_2,R_3) \leq H_D(R_0,R_1)+H_D(R_2,R_3)+ x * |X^jY^k|,$$
$$\text{where } x = 1 \text{ if } X^jY^k = B^lB^r,\ x = 0.5 \text{ if } X^jY^k \in \{ B^tB^b, T^tB^b, T^bB^b \}.$$

Proof: For brevity the proof is omitted. ☐

Before proving our main result in this section we establish a lower bound on the area required by an optimal solution. Our lower bound is given in table 4.1.

Lemma 4.2: Let D be an optimal assignment such that D' $\subseteq$ D. Assignment D and rectangles T and B satisfy the lower bound given in table 4.1.
Proof: The proof for these bounds is similar to the one in lemma 3.3. ☐

Theorem 4.1: For the 2_G-R2M problem, let D be an optimal assignment such that D' $\subseteq$ D and let M be the assignment generated by our algorithm. Then A(M) $\leq 2 *$ A(D).
Proof: For brevity the proof is omitted. ☐

V. APPROXIMATION ALGORITHM FOR THE 2-R2M PROBLEM.

In this section we show that our algorithm takes O(m log m) time and generates a solution with area $\leq 2 *$ OPT, where OPT is the area of an optimal solution.

```
algorithm for the 2-R2M problem
    Construct assignment D';
    Construct assignment $D_S$;
    Construct assignment $D_G(T^lT^r), D_G(B^lB^r), D_G(T^tT^b), D_G(B^tB^b), D_G(T^tB^t), D_G(T^bB^b), D_G(T^tB^b)$.
    Let P be the assignment generated by the above steps.
    Construct and output a layout with area A(P) for P.
end of algorithm
```

Theorem 5.1: The time complexity of our algorithm is O(m log m).
Proof: For brevity the proof is omitted. ☐

Theorem 5.2: Let D be an optimal assignment such that D' $\subseteq$ D and let P be the solution generated by our algorithm. Then, A(P) $\leq 2 *$ A(D).
Proof: For brevity the proof is omitted. ☐

VI. DISCUSSION.

We have shown that there is an efficient approximation algorithm that generates a solution within a factor of 2 of the optimal solution value for the 2-R2M problem. The algorithm takes O(m log m) time and the constant associated with this bound is small. In section I we assumed that no two terminals in the middle channel were placed along the same vertical line. If this is not the case, we may use four layers to route the set of nets T^bB^t by applying the optimal algorithm given in [12]. This procedure will not increase the time complexity of our algorithm. It is possible to obtain a better solution (not a better approximation bound) by using an optimal algorithm for the R1M problem to route the set of nets in M_G. However, for brevity this method will not be discussed.

VI. REFERENCE

[1] Baker, B. S. "A provable Good Algorithm for The Two Module Routing Problem", SIAM J. Computing 15:1, 1986, pp. 162-187.

[2] Gonzalez, T. and Lee S., "An Optimal Algorithm for Optimal Routing Around a Rectangle", Proceedings of the 20th Annual Allerton Conference on Comm, Control and Computing, U. of Illinois, Oct. 1982, pp636-645.

[3] Gonzalez, T. and Lee S., "An O(n log n) Algorithm for Optimal Routing Around a Rectangle", UTD Technical Report #116, Nov. 1982, (revised Oct. 1984).

[4] Gonzalez, T. and Lee S., "Routing Nets of Arbitrary Size Around a Rectangle", UTD Technical Report #202, Nov. 1983, (revised May 1985).

[5] Gonzalez, T. and Lee S., "A 1.6 Approximation Algorithm for Routing Multiterminal Nets," UTD Technical Report #204, May 1984, (revised May 1985), Also in Proceedings of the 22th Annual Allerton Conference on Comm, Control and Computing, U. of Illinois, Oct. 1984, pp636-645.

[6] Gonzalez, T. and Lee S., "Routing Multiterminal Nets Around a Rectangle", *IEEE Transactions on Computers*. Vol. C-35, No. 6, 1986 pp. 543-549.

[7] Gupta, U. I., Lee D. T. and Leung J., "An Optimal Solution for the Channel Assignment Problem", *IEEE Transactions on Computers*, Vol. C-28, No. 11, 1979, pp. 907-810.

[8] Hashimoto, A. and Stevens J. E., "Wire Routing by Optimizing Channel Assignment Without Large Apertures", Proceedings of the 8th IEEE Design Automation Conference, 1971, pp155-169.

[11] LaPaugh, A. S., "A Polynomial Time Algorithm for Optimal Routing Around a Rectangle", Proceedings of the 21st IEEE Foundation of Computer Science, 1980, pp282-293.

[10] LaPaugh, A. S., "Algorithms for Integrated Circuit Layout, An Analytic Approach", Ph.D. dissertation, Massachusetts Institute of Technology, 1980.

[11] Mead, C. and Conway, L., "Introduction to VLSI Systems", Addison-Wesley, 1980.

[12] Mehlhorn, k. and Preparata, F., "Routing Through a Rectangle" *Journal of ACM*, Vol. 33, No. 1, 1986, pp. 60-85.

[13] Pinter, R., "Optimal Routing in Rectilinear Channels", appeared in "*VLSI Systems and Computations*", Computer Science, pp. 160-177.

[14] Sahni, B., Bhatt, A. and Raghavan, R., "The Complexity of Design Automation Problems", Proceedings of the 19th Design Automation Conference, June 1981.

THREE LAYER CHANNEL ROUTING WITH STACKED PINS

MARTIN L. BRADY and DONNA J. BROWN
Coordinated Science Laboratory
University of Illinois
1101 W. Springfield Avenue
Urbana, IL 61801

We present bounds on channel width for a new three layer model for the channel routing problem (CRP). Previous channel routing models have restricted terminals to contain at most one net. We consider the three-layer restriction of a model introduced in [Sh86] in which CRPs may have *stacked pins*. In particular, we allow each terminal to contain up to two nets, entering on layers 1 and 3.

As more layers become available, allowing more than one wire at a terminal becomes desirable for some layout methodologies. For example, given three layers for standard cell layout, the third layer could be used to route over the rows of cells, eliminating the need for feedthroughs. This effectively defines a channel routing problem with at most two stacked pins at a terminal. One pin corresponds to a cell terminal (in the ordinary sense) and the second pin corresponds to a wire routed over the cell.

Although most previous two and three layer channel routing models have not allowed wires to *overlap* (i.e., run on top of each other), it is clear that stacked pins require a model in which some amount of overlap is allowed, at least in the vertical direction.

First, we consider *nonadjacent vertical overlap*, in which only vertical wires in layers 1 and 3 are allowed to overlap. The *density*, d, of the CRP is an obvious lower bound on the channel width. The channel width also depends on an extension of the term *flux*, defined in [BBL84] for (two-layer) Manhattan routing. The flux is the largest f such that a set of f^2 consecutive terminals in the CRP contains at least $2f^2 - f$ different nontrivial nets where we now define a trivial net as one having its upper and lower terminals in the same column *and layer*. Observe that since the set contains at most f unused pins, at most $f+2$ nets may be solved in track 1 (f nets may be solved within the set, and two nets may exit the set), $f+4$ in track 2, and so on. By the arguments of [BR81], [BBL84], at least f tracks are required overall, leading to the following lower bound.

> *Theorem 1.* The minimum channel width for a stacked pin CRP with density d and flux f in the three-layer nonadjacent vertical overlap model is at least $max(d, f)$.

Notice that a CRP in which all nets have their two terminals in the same column but in different layers has density 0, but the flux is $\Theta(\sqrt{N})$, where N is the number of nets in the CRP.

The similarity of this model to Manhattan routing includes the upper bound as well. Even if we restrict the model to a reserved layer "VHV" scheme, where layers 1 and 3 contain only vertical wires and layer 2 contains only horizontal wires, the 2-layer algorithm of [BBL84] can be extended to the 3-layer stacked pin model. The basic idea is to transform a two layer Manhattan routing by "squeezing" a pair of consecutive columns into a single column having four terminal connections. Two connections are routed in layer 1 and the other two are routed in layer 3, giving the following upper bound.

> *Theorem 2.* Any 2-terminal net stacked pin CRP with density d and flux f can be routed in the three-layer VHV model using $d + O(f)$ tracks.

Next we consider a small relaxation of the nonadjacent vertical overlap model, called the *unit-length triple vertical overlap* model. We allow wires to overlap in all three layers only for unit-length vertical segments (in addition to allowing vertical overlap of unrestricted length between layers 1 and 3). This small relaxation yields a definite improvement in the channel width. In particular, the ideas presented in [BBBL85], [BraBr85] can be extended to give the following upper bound, not dependent upon flux.

Theorem 3. Any 2-terminal net stacked pin CRP with density d can be routed in the three-layer unit-length triple vertical overlap model using $d + O(\sqrt{d})$ tracks.

It seems that the unit-length triple vertical overlap stacked pin model corresponds to the 2 layer unit vertical overlap model. Once again, we note that a pair of adjacent columns in the 2 layer algorithm corresponds to a single column with four terminal connections in the stacked pin model. This result could not be greatly improved because even if any length of triple vertical overlap is allowed, at least $d + \Omega(\log d)$ tracks are required, by extension of the 2-layer lower bound found in [BBBL85].

Theorem 4. Stacked pin CRPs of density d exist which require $d + \Omega(\log d)$ tracks to route in the triple vertical overlap model.

The above results are summarized in the following table:

Model	Lower Bound	Upper Bound
Nonadjacent vertical overlap	$max(d, flux)$	$d + O(flux)$
VHV	$max(d, flux)$	$d + O(flux)$
Unit-length triple vertical overlap	$d + \Omega(log\ d)$	$d + O(\sqrt{d})$
Triple vertical overlap	$d + \Omega(log\ d)$	$d + O(\sqrt{d})$
Unrestricted triple overlap	d^*	$d + O(\sqrt{d})$

*This bound was obtained by [Sh86].

Table 1. Summary of 3-layer stacked pin bounds.

References

[BBL84] Baker, B., S. N. Bhatt, and T. Leighton, "An Approximation Algorithm for Manhattan Routing," *Advances in Computing Research* 2 *(VLSI Theory)*, ed. F. P. Preparata, JAI Press, Inc., Greenwich, CT, pp. 205-229 (1984).

[BBBL85] Berger, B., M. Brady, D. J. Brown, and T. Leighton, *Nearly Optimal Algorithms and Bounds for Multilayer Channel Routing*, draft, 1986.

[BraBr85] Brady, M. and D. J. Brown, "An Algorithm for Three Layer Channel Routing Using Restricted Overlap," *23rd Annual Allerton Conference on Communication, Control, and Computing*, pp. 674-675 (Oct. 1985).

[BR81] Brown, D. J. and R. L. Rivest, "New Lower Bounds for Channel Width," *Proc. 1981 CMU Conf. on VLSI Systems and Computations*, pp. 178-185 (Oct. 1981).

[Sh86] Shaffer, R., "Multilayer Channel Routing with Stacked Terminals," Masters Thesis, University of Colorado-Boulder (May 1986).

The Energy Complexity of Transitive Functions [1]

LAWRENCE SNYDER & AKHILESH TYAGI
Department of Computer Science
University of Washington
Seattle, WA 98195

Abstract

We define a *normal* transitive function to be a function that embeds a computation of the permutation group generated by the cycle $(1\ 2\ 3 \ldots n)$. Despite this restriction, the class of normal transitive functions is rich enough to include all the functions that Vuillemin showed to be transitive. A partial list consists of shifts, cyclic shifts, multiplication, convolution and linear transform. We show that for an implementation of these functions, where every wire is allowed to switch at most once, the average switching energy, $E_a(C)$, is $\Omega(n^2)$. If a wire is allowed to switch more than once then we prove a lower bound of $E_a(C) = \Omega(n^{3/2})$. We prove that ***word systolic systems*** for transitive functions consume the same order of energy both on average and in worst case. In particular, the average case energy $E_a(C)$ is $\Omega(A)$ and the worst case energy $E_w(C)$ is $O(A)$, where A is the area of any embedding for a circuit C to compute a transitive function. However, to demonstrate that this unexpected behavior is not universal, we show the existence of a systolic system for a problem for which the average case and worst case energy consumptions can be separated. We also extend a result by Kissin on a lower bound on the worst case switching energy of 1-switchable functions. For any convex embedding of a circuit to compute a 1-switchable function within depth $d(n)$, $log^2 n \leq d(n) \leq n^{\epsilon}$, with $0 < \epsilon \leq 1$, $E_w(C) \times d(n)$ is $\Omega(\max(n \log n,\ n\, d(n))$. Note that every transitive function is also 1-switchable, implying that these lower bounds apply to transitive functions, as well.

1 Introduction

Due to engineering limitations on heat dissipation from a planar chip and a general trend towards energy conservation, energy efficiency of VLSI circuits has become an important issue in VLSI algorithm design. Despite relatively low energy consumption of CMOS technology, the energy dissipation is still an issue in the design of high performance systems for the following reasons. The circuit designers tend to overdrive the CMOS devices for higher speeds. Worst still, in CMOS technology, the power consumption is a direct function of the system frequency. Thus high performance architectures need to be designed even more carefully, to be energy conscious. Most high performance systems are based on the concept of pipelining or systolic dataflow. The trend towards integrating most of the architectural components onto fewer and fewer chips has encouraged the design of datapath and control components with systolic algorithms due to their simple communication structure. Vuillemin [10] identified an important class of vfunctions known as *transitive* functions. Interestingly, this class encompasses all the data path functions like multiplication and shifting. Thus, our result has a wide impact in as much as it frees the designer from considering energy optimization as one of the objectives of the design. The second result reconfirms the widely held belief that the designer can trade the speed of operation with the energy consumption.

With the wider availability of VLSI design environments, and introduction of systematic design methodology by Mead and Conway [7], relatively inexperienced designers are transcribing their algorithms into silicon. Often, such engineering applications include multiplication, convolution and linear transforms. Interestingly, all these functions are normal transitive functions.

---

[1] Supported in part by DARPA under Contract MDA903-85-K-0072

Lengauer and Mehlhorn [6] have shown that for a function with $AP^2 = O(n^2)$, switching energy is bounded from below by $\Omega(AP)$, where A is the area and P is the period of a pipelined computation. Kissin [5] proved that for some monotonic circuits, switching can be superlinear, but switching energy is still bounded by $O(A)$. Kissin [4] explores switching complexity of *comparison, or* and *addition* functions. She shows that for 1-switchable functions, $\Omega(n \log n)$ worst case switching energy is required if computation is to be performed within depth $O(\log n)$. She also gives a linear average energy layout for an adder.

We first describe the VLSI model of computation and energy consumption. We prove a lower bound of $\Omega(n^2)$ on the average switching energy of normal transitive functions. In section 4, we show that the lower bound on average switching energy of a transitive systolic system matches the upper bound on the worst case switching energy. We conclude with the result about depth switching-energy trade-off. For a detailed version of this work, the reader is referred to the technical report [8].

2 Model

We will formally define the model in which we charge for energy. Our energy model is the same as the one originally outlined by Kissin [5]. For the sake of completeness, we will briefly describe the model. The VLSI model is the commonly accepted one, proposed by Thompson [9]. A layout can be viewed as an embedding of the communication graph in a Cartesian grid. Each grid point can either have a processor or a wire passing through. A wire can not go through a grid point unless it is a terminal of the processor at that grid point. Wires have unit width and bandwidth and processors have unit area. The initial data values are localized to some constant area, to preclude an encoding of the results.

Let u be the unit switching energy defined to be the energy spent when a wire of unit length switches (changes state either from a $1 \rightarrow 0$ or from a $0 \rightarrow 1$). Note that a minimum size processor also consumes $O(u)$ energy when it changes state. We will say that a wire of length l consumes $\Omega(l)$ energy when it switches. It is implicit here that u is a technology dependent constant.

We do not account for the switching energy consumed by the processors. This is in accordance with the widely held belief that the wires take up most of the area in a VLSI layout. Since we are working with the lower bounds on switching energy, energy consumption of the wires definitely provides a lower bound on the total energy consumed.

We work with the Uniswitch Model (USM), as defined in Kissin [5]. In this model, a signal can propagate along a wire of arbitrary length in constant time. This restricts every wire in an acyclic circuit to switch at most once, for unpipelined computation. A symmetric notion is that of the Multiswitch Model. Kissin shows some upper bounds on switching energy of certain circuits in the Multiswitch model.

ASSUMPTIONS:

1. Each node in the circuit depends on an input, *i.e.*, for each node, a pair of inputs exists which makes that node switch.
2. All the inputs are synchronous. In other words, they are applied at the input ports, simultaneously.
3. Circuits are synchronous.

We denote a circuit $C(V, W)$ as a graph, where V is the set of nodes and W is the set of edges or wires in $V \times V$.

Definition 1 *A circuit $C(V,W)$ is said to be in state $s : V \cup W \longrightarrow \{0,1\}$, if s is consistent according to the following conditions.*

- *For an input node x_i, $s(x_i)$ is consistent with the input $x_0x_1 \ldots x_{n-1}$. For an input wire $w = (x_i, y)$, $s(w)$ must equal $s(x_i)$.*
- *Non input nodes and edges have the values consistent with the input values and the labels of the nodes. For example, for a node, v, labelled by $\wedge$ with state of input wires $s(w_1) = 1, s(w_2) = 1$, $s(v)$ must equal 1.*

Definition 2 *A wire w (node v) is said to have switched from state s_0 to state s_1 if $s_0(w) \neq s_1(w)$ $(s_0(v) \neq s_1(v))$.*

We define a measure of energy consumption for a circuit. When a circuit C is subjected to an input $\vec{x}$, let wire w_i switch k_i times before the circuit is settled. Let l_i be the length of a wire w_i in the circuit C's embedding in a grid. Then the energy consumption for circuit C, $E_W(C, s, \vec{x})$ in state s with $\vec{x}$ as the input is defined to be $u \times \sum_{w_i \in W} k_i \times l_i$. For the Uniswitch model $k_i \leq 1$. We will distinguish between worst case energy consumption and average case energy consumption.

Definition 3 *The worst case energy consumption for a circuit C, $E_w(C)$, is defined to be $max_{s,\vec{x}} E_W(C, s, \vec{x})$, where max is taken over all (state, input vector) pairs.*

We similarly define the average case energy consumption. Note that an input assignment $\vec{z}$ defines the state of the circuit completely.

Definition 4 *The avarage case energy consumption for a circuit C is defined to be its energy consumption averaged over all initial states and all input vectors. Thus $E_a(C) = \sum_{s,\vec{x}} E_W(C, s, \vec{x}) \,/\, 2^{2n}$, where n is the number of input bits.*

For further details of this model, the reader is referred to Kissin's papers [4], [5].

3 Normal Transitive Functions

In this section, we show that for a general implementation of a normal transitive function (defined below), the average case energy $E_a(C)$ is $\Omega(n^2)$. The class of normal transitive functions encompasses all the functions that were used as examples of transitive functions by Vuillemin [10]. We first show that for a normal transitive function, $E_a(C)$ is no less than the switching energy consumed by $n/2$ edge disjoint paths from input ports to the output ports. Then we show that the energy consumed by these $n/2$ paths is $\Omega(n^2)$.

We first define a class of functions analogous to Vuillemin's transitive functions.

Definition 5 *A boolean function $f(x_1, x_2, \ldots, x_n, c_1, c_2, \ldots, c_k) = (y_1, y_2, \ldots, y_n)$ is said to be normal transitive, if it computes either the permutation group generated by the cycle $(1\ 2\ 3 \ldots n)$ or a product of such groups. The bits $c_1, c_2, \ldots, c_k$ are the control bits specifying a permutation group element.*

In other words, the *normal transitive functions* embed a shifting like computation as a specialized instance. It is easy to verify that all the functions Vuillemin shows to be transitive, are just normal transitive. All his proofs show a reduction to the shifting permutation group. In particular, shifts, cyclic shifts, multiplication, convolution, linear transform and three matrix multiplication are all normal transitive functions.

We exploit the structure of the shift permutation group to show that when each of the n input bits and c control bits switch independently with probability 1/2 each, the expected number of output bits that switch is $n/2$.

Theorem 1 *For a normal transitive function, on average $n/2$ output bits switch.*

PROOF SKETCH: We prove the result for a shifting permutation group. It can be easily extended to a product of two shifting permutation groups of different order.

Let x_j denote the jth input bit and let y_l denote the lth output bit. Without loss of generality, assume that at time t, the control bits specify the identity permutation, *i.e.*, $y_j(t) = x_j(t), 1 \leq j \leq n$. Since f computes the shift permutation group, for any $\pi \in G$ such that the encoding of π differs from encoding of the identity permutation by $k/2$ bits, $y_{(j+k)\mathrm{mod} n}(t+1) = x_j(t+1), 1 \leq j \leq n$, for some $0 \leq k \leq n-1$. In other words, each input bit is connected to an output bit k bits up/down. $n/2$ of n input bits are expected to switch. There are two cases.

1. $x_j(t) = x_{(j+k)\mathrm{mod} n}(t)$. In this case, $x_j(t) = y_{(j+k)\mathrm{mod} n}(t)$. The probability that $x_j(t)$ does not equal $x_j(t+1)$ is $1/2$. This implies that the probability that the bit $y_{(j+k)\mathrm{mod} n}(t) \neq y_{(j+k)\mathrm{mod} n}(t+1)$ is $1/2$.
2. $x_j(t) \neq x_{(j+k)\mathrm{mod} n}(t)$. This case is similar to the previous one. Here, $x_j(t) \neq y_{(j+k)\mathrm{mod} n}(t)$. The probability that $x_j(t)$ does not equal $x_j(t+1)$ is $1/2$. Once again, this means that the probability that the bit $y_{(j+k)\mathrm{mod} n}(t) \neq y_{(j+k)\mathrm{mod} n}(t+1)$ is $1/2$.

This shows that the probability of each output bit switching independently is $1/2$. Hence the number of output bits expected to switch is $n/2$.

□

The previous theorem shows that half of n output bits are expected to switch, even when we allow the permutation group element to change. Let $\mathcal{I}$ and $\mathcal{O}$ be the sets of input and output bits respectively. A partition of the chip is denoted by $p = (\mathcal{I}_L, \mathcal{I}_R, \mathcal{O}_L, \mathcal{O}_R)$, where $\mathcal{I} = \mathcal{I}_L \cup \mathcal{I}_R$ and $\mathcal{O} = \mathcal{O}_L \cup \mathcal{O}_R$. Let (i, o), where $i \in \mathcal{I}$, $o \in \mathcal{O}$, be an input bit position, output bit position pair. A pair (i, o) is said to be a *straddled pair* if either $i \in \mathcal{I}_L$ & $o \in \mathcal{O}_R$ or $i \in \mathcal{I}_R$ & $o \in \mathcal{O}_L$. There are $n/2$ (i, o) pairs that switch. In order to be able to show that most of the area of a chip switches, we have to show that a significant fraction of $n/2$ switching (i, o) pairs are straddled pairs.

Lemma 1 *For a normal transitive function, at least $n/3$ of n (i, o) pairs are straddled pairs.*

PROOF SKETCH: A partition could either divide $\mathcal{I}$ relatively evenly between two sides, *i.e.*, both $1/3 < |\mathcal{I}_L|, |\mathcal{I}_R| \leq 2/3$; or one of $\mathcal{I}_L$ or $\mathcal{I}_R$ has more than $2n/3$ bits. In the second case, it is easy to see that at least $n/3$ of the input bits from the side with $> 2n/3$ input bits will be paired with the output bits on the opposite side. In the first case, we average over all the n control word values. Let c denote the value of control bits $c_{\log n} \dots c_2 c_1$. Let $\delta_{i,c}$ be 1 if x_i and y_{j+c} are on the opposite sides, and 0 otherwise. For every output bit y_j, there are at least $n/3$ input bits which are on the other side. Thus $\sum_{c=0}^{n-1} \delta_{j-c,c} \geq n/3$. If we sum it over all the output bit positions j, and divide it by the number of permutation group elements n, we find that an average group element creates at least $n/3$ straddled (i, o) pairs.

□

By Theorem 1, at least half of these straddled (i, o) pairs are expected to switch. Thus there are at least $n/6$ input bits that switch and whose corresponding output bits are on the opposite side. Next step is to show that these $n/6$ straddled pairs switch most of the area.

Theorem 2 *Let $f(x_1, x_2, \dots, x_n, c_1, c_2, \dots, c_{\log n}) = (y_1, y_2, \dots, y_n)$ be a normal transitive function. The switching energy consumed by the switching of $\Omega(n)$ (i, o) pairs (x_{i_1}, y_{j_1}) $(x_{i_2}, y_{j_2}) \dots (x_{i_{kn}}, y_{j_{kn}})$ is given by $\Omega(n^2)$.*

PROOF SKETCH: The proof is similar to the one given by Thompson to show that the area of a graph with bisection width ω is $\Omega(\omega^2)$. The bisection width of the subgraph spanned by $\mathcal{I}$ and $\mathcal{O}$ is k. We count the contribution to the lengths of the edges of k disjoint paths crossing a bisecting line $x = \alpha$ with one jog. Thompson counts the area contribution of the edges crossing such a line. □

We can generalize this result to the circuits where a wire is allowed to switch more than once. Then a circuit to compute a normal transitive function could have a bisection width $1 \leq k \leq n$. Each of the k edges in the bisection will have to multiplex $\Omega(n/k)$. We have to preclude the possibility that the chip sorts n/k bits in each group and sends them in that order, thus making the long wires switch only once. The following lemma shows that, on average, a n/k bit sequence has about $n/2k$ bit alternations. We assume that the sequencing of the bits in a n/k bits group is oblivious to the value of the input bits. We say that in a bit sequence $a_1, a_2, \ldots, a_l$, there is an alternation at the position j if $a_j \neq a_{j+1}$ for $1 \leq j \leq l-1$, *i.e.*, δ_j equals 1 if $a_j \neq a_{j+1}$ and 0 otherwise. The *total alternation* for a l-bit sequence is given by $\sum_{j=1}^{l} \delta_j$.

Lemma 2 *The average total alternation for a k-bit sequence from the set* $\{0,1\}^k$ *is* $(k-1)/2$.

PROOF SKETCH: Let the total alternation summed up over all the bit sequences in $\{0,1\}^k$ be denoted by $A(k)$. $A(k)$ is given by the following recurrence.

$$A(k) = 2A(k-1) + 2^{k-1};\ A(1) = 0;$$

This equation has a solution in $A(k) = (k-1)2^{k-1}$. Averaged over 2^k k-bit sequences in $\{0,1\}^k$, we get an average total alternation of $(k-1)/2$.

□

By the preceding lemma, each of k edge disjoint paths will switch $\Omega(n/k)$ times. The following theorem shows that most of the area will switch every time.

Theorem 3 *The average switching energy of a normal transitive function computed by a circuit of bisection width k is bounded from below by* $\Omega(kn + n^2/k)$.

PROOF SKETCH: The proof to show that $\Omega(k^2)$ switching takes place at each time step is exactly similar to the proof of Theorem 2. When k is $O(1)$, just the storage requirements give $\Omega(n)$ switching energy. This adds up to $\Omega(k^2 + n)$ switching energy at each step. Lemma 2 assures that there will be $\Omega(n/k)$ time steps due to bandwidth limitations. The lower bound of $\Omega(kn + n^2/k)$ is given by multiplying the lower bounds on area and time steps derived above.

□

The lower bound on the average switching energy of a normal transitive function derived in Theorem 3 attains a minimum of $n^{3/2}$ when the bandwidth is $\sqrt{n}$.

4 Energy Complexity of Transitive Systolic Systems

In this section, by *Transitive Systolic System* we will refer to a system

- that is systolic.
- which computes a transitive function.

There are many versions of what it means for an algorithm to be systolic. We attempt to capture most of the commonly accepted characteristics of a systolic system in the following discussion.

1. *regular structures*: The most distinguishing characteristic of systolic systems is their regular structure. This makes them very suitable for VLSI implementations.

2. *few neighbors*: Each processing unit has a constant number of neighbors, which is independent of the input size n.

3. *pipeline*: These structures are capable of supporting a pipeline of input data at a regular pipeline period. Without loss of generality, we assume that at each time unit, a systolic system produces an output bit for each output data stream. The bubbles in a stream can be accommodated by defining a larger clock tick, within which the system is internally introducing these bubbles. Alternatively, one could distinguish between *bit systolic* and *word systolic* systems. A systolic system is *word systolic* if it produces a *word* of output at each clock tick. Similarly, a systolic system is *bit systolic* if it produces a *bit* of an output word at each clock tick. We insist on *word systolicity*.

4. *distribution of streams*: The input and output data streams should be evenly distributed around the periphery. This requirement is imposed by our desire to be able to use this structure in a larger system with compact routing. In an extreme case, if all the input and output data streams were tapped from the same convex side of the system, chances are that there will be a pitch mismatch between I/O side of this unit and I/O sides of other units overlayed together.

5. *nonzero delay*: Each stream has delay at least one in each processor.

We briefly describe what a transitive function is. We borrow the notation from Vuillemin [10].

Definition 6 *A boolean function* $f(x_1, x_2, \ldots, x_n, c_1, c_2, \ldots, c_k) = (y_1, y_2, \ldots, y_n)$ *is said to be transitive if:*

1. *For each value of the control input bits* $c_1, c_2, \ldots, c_k$ *the output vector* $(y_1, y_2, \ldots, y_n)$ *is the permuted input vector* $(x_{g(1)}, x_{g(2)}, \ldots, x_{g(n)})$*, where the permutation* g *is determined by the control bits.*

2. *The set of elements* g *forms a transitive permutation group.*

3. *For each pair* (i, j) *of index positions,* $1 \leq i, j \leq n$*, there exists a permutation* π*, such that* π *maps* i *into* j*, i.e.,* $\pi(i) = j$*.*

The main property of transitive functions is that each output bit depends on each input bit. Note that this makes transitive functions more restrictive than the functions for which each output value depends on each input value.

A systolic system may not exist for every transitive function. A function could be arbitrarily complex and still be transitive. An example is the union of a nonrecursive function and shift function, which is not even solvable. But, it still embeds an instance of a transitive computation. This result applies to only those transitive functions that are simple enough to have a systolic algorithm.

Before we go any further, we develop some notation. A data stream is said to be an input(output) data stream if the flow of data through this stream is into(out of) the system. Each input stream is incident on two convex faces of the convex embedding of the system. For a stream D, let $IF(D)$ denote the input face and let $OF(D)$ denote the output face.

Definition 7 *A description of a data stream* D*,* $des(D)$*, is a list* $w_0\, pe_1\, w_1 \ldots pe_i\, w_i \ldots pe_n\, w_n$ *of alternating wire segments and processors, such that wire segment* w_{i-1} *is incident into and wire segment* w_i *is incident out of the processor* pe_i*.*

Definition 8 *An input data stream D_i is said to dominate an output data stream D_o, denoted by $D_i \vdash D_o$, if both D_i and D_o are incident on a processor where bits in D_i are input to the computation of bits of D_o.*

Lemma 3 *For a transitive systolic function, every input data stream dominates every output data stream.*

REMARK 1: The description of a data stream D, $des(D)$ has a processor, wire segment pair (p_j, w_j), for every output data stream OD_i that D *dominates.*

Definition 9 *The span of an input data stream is defined to be the number of output data streams it dominates.*

Lemma 4 *In a transitive systolic system, when an input bit is switched, $\Omega(n)$ switching energy is consumed.*

PROOF SKETCH: Let us consider any input data stream, ID_i. Let E_i be the switching energy consumed when the data stream ID_i switches. We know from Lemma 3 that ID_i dominates each output data stream OD_j for $1 \leq j \leq m$. As we mentioned in Remark 1, this implies that $des(ID_i)$ has a processor, wire segment pair (p_j, w_j) for each of the output data streams OD_j. Thus, $des(ID_i)$ has at least $m = \Omega(n)$ wire segments.

According to our model, each wire segment has length $\Omega(1)$. Each wire segment w_k, when switched, consumes $e_k = \Omega(1)$ energy. When the input stream ID_i switches, each of the wire segments w_k in its description $des(ID_i)$ will switch. Thus, E_i, the energy consumed when ID_i switches is $\Omega(\sum_{w_k \in des(ID_i)} e_k)$. Since each e_k is at least constant and $des(ID_i)$ has $\Omega(n)$ wire segments, E_i is $\Omega(n)$.

□

Now we will prove a series of lemmas that lead us to our main theorem. In what follows, n is the number of input bits and m is the number of the output bits for the transitive function under discussion.

Lemma 5 *For a transitive systolic system, no input data stream can multiplex $o(k)$ input bit positions, for a constant c.*

PROOF SKETCH: The proof is based on the fact that for computing any output bit in a transitive function, every input bit is required. Each output bit requires n input bits. Thus each input data stream should provide one bit at each clock tick.

□

The key idea on which the proof of main theorem hinges, is that the area of a transitive systolic system is $\Theta(m \times n)$. Since m is $\Theta(n)$, this means that A is $\Theta(n^2)$. We first show that A is $\Omega(m \times n)$.

Lemma 6 *The area A of a transitive systolic system is bounded from below by $\Omega(m \times n)$.*

PROOF SKETCH: Notice that in any systolic system, no output bit stream can be multiplexed. Otherwise, there will be a constant buildup of information within the system. None of it can be discarded, since a transitive function needs it all. By Lemma 4, for a transitive function every input stream has a span of $\Omega(m)$. Thus each input stream induces wire length of $\Omega(m)$ in the system. By Lemma 5, there are at least $\Omega(n)$ such wires. This proves the result. □
Now we are ready to show that A is $O(m \times n)$.

Lemma 7 *For transitive systolic functions, the area A of the system embedding is $O(m \times n)$.*

PROOF SKETCH: If a systolic system exists for a function, then an equivalent one can be realized as in the schema shown in Figure 1. The size of each processor P is a constant p. If it were not, then the system would not be systolic. Since the number of input bits and output bits is a constant, there is only a finite number of Boolean functions realizable. Each of these functions can be realized in constant area. This proves the result.

□

Theorem 4 *For transitive systolic systems, the average energy consumption is of the same order as the worst case energy consumption.*

PROOF SKETCH: Let $C = (V, W)$ be the circuit to realize a transitive function f with a systolic algorithm. The worst case (state, input) combination can make all of the area switch. Thus $E_w(C)$ is $O(A)$.

For a uniform distribution, each input bit x_i to the circuit can switch independently. The probability that a bit x_i switches $\mathrm{P}[x_i \text{ switches}] = \mathrm{P}[x_i \text{ goes } 0 \rightarrow 1] + \mathrm{P}[x_i \text{ goes } 1 \rightarrow 0] = 1/2$. Thus the expected number of input bits switching is $n/2$. By Lemma 4 each switched input bit consumes $\Omega(n)$ energy. Thus a lower bound on $E_a(C)$ is $\Omega(n^2)$. From Lemmas 6 and 7, area A is $\Theta(n^2)$. This shows that $E_a(C)$ is $\Omega(A)$ and $E_w(C)$ is $O(A)$.

□

In the next section, we demonstrate this result with an example of a multiplier circuit. This algorithm is due to Wayne Winder of the VLSI Consortium at the University of Washington. This algorithm is of independent interest too, since it achieves the lower bound of n^2 on AP^2 for multiplication, as shown by Vuillemin [10].

4.1 A Multiplier Example

This multiplier is based on the basic shift and add paradigm of multiplication. The multiplication algorithm that we all learnt in grade school is shown in Figure 2. If instead of writing row i shifted left by $i - 1$ for $1 \leq i \leq m$, we wrote them all aligned, we will get this multiplication scheme. Each cell (i, j) receives a carryin from its north neighbor and a sum bit s_j from its northwest neighbor. It sends out the sum of $b_i \times a_j$, s_j and the carryin to its southeast neighbor, and carryout to its south neighbor. The last row is an exception. To get output bits $c_{n/2}$ on the right, we have an adder that sums up previous carrys with the sum bits. We can use an adder proposed by Brent and Kung [2]. This adder propagates the carry in time $O(\log n)$ and with width $O(n)$ and height $O(\log n)$. When pipelined, the adder has a period of $O(1)$. Thus the whole multiplier has area $O(n^2)$ and it works at a pipeline period of $\Theta(1)$. Thus its AP^2 is $O(n^2)$. Note that each input data stream has a total wire length equal to $\Omega(n)$. On average half the input bits switch, giving us a an avarage case energy consumption of $\Omega(n^2)$. The area of this circuit is $\Theta(n^2)$. The worst case switching energy is bounded by $O(A)$. This shows that this multiplier indeed consumes the same order of average case and worst case switching energy.

4.2 CFL Example

Just to demonstrate that it is not always true, we separate the average and worst case energy consumptions of a systolic implementation for recognizing a context free language $L = \{ww^R\} \cup \{1\{0,1\}^*\}$. We refer to the systolic implementation of Guibas, Kung and Thompson [3]. In the worst case, a string of the form ww^R where $w \in \{0,1\}^n$ dominates with an energy complexity of n^2. In the average case, though, the strings of the form $1\{0,1\}^{n-1}$ dominate, since there are $2^{2n-1} - 2^n$ of them compared to 2^n of ww^R kind.

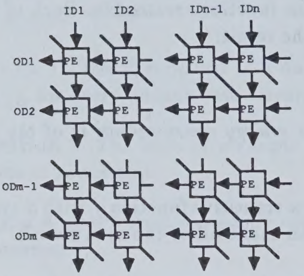


Figure 1: The Systolic System Schema

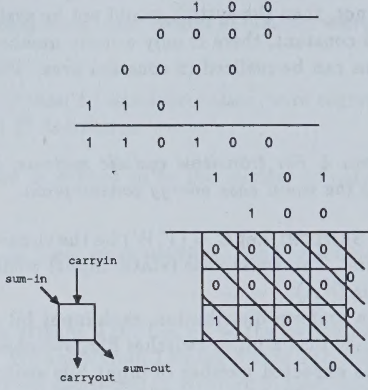


Figure 2: The Multiplication Algorithm

5 Lower Bounds

In this section, we extend the lower bound result of Kissin [4]. She shows that to compute a 1-switchable function in depth $O(log\,n)$, worst case switching energy of $\Omega(n\,log\,n)$ must be used. The assumption is that the output nodes are placed on a convex border in the embedding. We show that if the depth can be relaxed to be a polynomial in n, *i.e.*, $d(n) = n^\epsilon$ for $0 < \epsilon \leq 1$, then area × depth $= \Omega(\max(n\,log\,n,\ n^{1+\epsilon}))$. Similarly, if the depth is polylog, *i.e.*, $d(n) = log^k n$ for $k \geq 2$, then area × depth $= \Omega(n\,log^k n)$. These lower bounds hold under the assumptions that in an embedding of the circuit C the output nodes are located on a convex border and the fan out is limited to two.

Next, we will define *1-switchable functions.* A function f is 1-switchable if there is an input bit position on which every output bit depends. A formal definition follows.

Definition 10 *A function* $f : \{0,1\}^n \longrightarrow \{0,1\}^m$*, with* $m = \Theta(n)$*, is said to be 1-switchable if there exist two input vectors* $\vec{a} = x_1x_2\ldots x_k\ldots x_n$ *and* $\vec{b} = x_1x_2\ldots\overline{x}_k\ldots x_n$ *differing only in one bit position, such that* $\Theta(n)$ *output bits switch from* $f(\vec{a})$ *to* $f(\vec{b})$.

Note that the class of transitive functions forms a subset of the class of *1-switchable* functions. Thus all the lower bounds that we derive also apply to transitive functions.

Our basic problem is to fan out the input bit x_k to $\Theta(n)$ output bit positions. If we restrict the depth for this fan out, then we can place lower bounds on the sum of the lengths of the edges in a Cartesian grid embedding. We limit ourselves to the embeddings (underlying communication graphs) with fan out of at most 2 for every vertex. Our strategy will be to first show that any efficient fan out graph will be in a canonical form, which we call *balanced fan out* graph. Then we show a lower bound on the area of any embedding for this graph, given a certain depth. Note that since we are working with the unit delay model, depth corresponds to delay in the given function. A wire can be driven in unit time regardless of its length.

Definition 11 *Let* $G = (V, E)$ *be a directed, rooted graph. Let* $v_O \in V$ *be the designated input node and* $V_O \subseteq V$ *be a designated set of output nodes Then* $F = (V, E, v_O, V_O)$ *is said to be a fan out graph if there exists a directed path in* G *from* v_O *to each* $v_k \in V_O$.

We limit ourselves to only those embeddings in which the output nodes are located at a convex border. Without loss of generality, we can assume that all the output nodes are located along a straight line. This blows up the area at most by a constant factor. Thus we restrict the definition of a fan out graph to get an appropriate class of graphs which admit such convex embeddings. In a digraph $G = (V, E)$ let u be the predecessor of v, denoted by *pred*(v), if the edge $(u, v) \in E$. In this case, v is called u's successor, denoted by *succ*(u).

Definition 12 *A fan out graph $F = (V, E, v_0, V_O)$ is said to be a convex fan out graph if for every $v \in V_O$, its successor also is an output node, succ$(v) \in V_O$. In this case, all the nodes in $V - (\{v_0\} \cup V_O)$ are called internal nodes.*

Definition 13 *A convex fan out graph $F = (V, E, v_0, V_O)$ is said to be optimal if there is no other convex fan out graph from v_0 to V_O with depth less than equal to depth(F), and with an embedding with less area than minimum area embedding of F.*

REMARK 2: Any optimal convex fan out graph will be acyclic. Back edges do not fan out into a new node. There is a convex fan out graph with all the back edges removed, which has the same order of depth but less switching energy. Now on, we will assume that our canonical form graphs, balanced fan out graphs, are acyclic.

Definition 14 *A linear chain consists of an alternating node,edge list, $v_i\, e_i\, v_{i+1} \ldots v_{j-1}\, e_{j-1}\, v_j$, such that $e_k = (v_k, v_{k+1})$, for $i \leq k \leq j-1$, and every v_l, in the list has at most one outgoing edge. A linear chain is maximal if pred(v_i) has two outgoing edges and either v_j is a leaf or succ(v_j) has two outgoing edges. Node v_i is called the leader of the chain. The length of the chain is the number of nodes in its list.*

Lemma 8 *An optimal convex fan out graph does not have any maximal linear chains consisting of internal nodes.*

PROOF SKETCH: By contradiction. Assume that $F = (V, E, v_0, V_O)$ is optimal and it has a linear chain. $v_i\, e_i\, v_{i+1} \ldots v_{i+l} \ldots e_{i+k-1}\, v_{i+k}$ (note that if v_{i+l+1} is an output node, then no node $v_q, i \leq q \leq i+l$ can be an output node and no node $v_r, i+l+1 \leq r \leq i+k$ can be an internal node, since F is a convex fan out graph). We can get a new convex fan out graph, F', by deleting nodes v_i through v_{i+l} and edges e_{i-1} through e_{i+l} and by inserting the edge (*pred*$(v_i), v_{i+l+1}$). Note that F' has depth at most equal to the depth of F and its switching energy has been reduced by at least $l+1$.

□

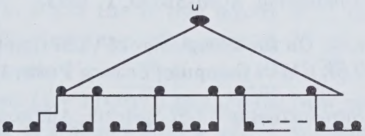


Figure 3: Canonical Form Embedding

NOTE: A linear chain consisting of only output nodes is called a *terminal chain.*
Thus , an optimal convex fan out graph will consist of a binary tree stage whose leaves each feed a linear chain, as shown in Figure 3. It is easy to see that, for optimality, the binary tree must be a balanced binary tree. It will be a complete binary tree if the number of leaves is an integral power of two.

Definition 15 *A set of linear chains is balanced if every chain has length k or $k+1$ for some integer k.*

Lemma 9 *For an optimal convex fan out graph, the set of its terminal chains is balanced.*

This defines our family of acceptable fan out graphs. A fan out graph F is said to be a *balanced fan out graph* if it is a convex acyclic fan out graph with balanced terminal chains.

Theorem 5 *Let* $f : \{0,1\}^n \longrightarrow \{0,1\}^m$ *be a 1-switchable function. Let* $C(V, W)$ *be a circuit to compute* f. *For any convex embedding of* C *within depth* $d(n)$, $log^2 n \leq d(n) \leq n^\epsilon$, *with* $0 < \epsilon \leq 1$, *the following holds. The* $E_w(C) \times d(n)$ *is* $\Omega(\max(n \log n,\ n\, d(n))$.

PROOF SKETCH: The proof relies on the result of Brent and Kung on the minimum area required for a convex embedding of a binary tree [1]. The terminal linear chains along a line contribute $\Omega(n)$ to the sum of edge lengths, and consequently to area and switching energy. The second factor of $n/d(n) \times log\,(n/d(n))$ is from the balanced binary tree embedding.

□

6 Acknowledgements

We are indebted to the staff of the University of Washington VLSI Consortium in many ways, particularly Larry McMurchie and Wayne Winder. We thank Wayne Winder for letting us use his multiplication algorithm as an example.

References

[1] R.P. Brent and H.T. Kung. The Chip Complexity of Binary Arithmetic. In *ACM Symposium on Theory of Computing*, ACM-SIGACT, 1980.

[2] R.P. Brent and H.T. Kung. A Regular Layout for Parallel Adders. *IEEE Transactions on Computers*, March 1982.

[3] L. J. Guibas, Kung H. T., and Thompson C. D. Direct VLSI Implementation of Combinatorial Algorithms. In *Cal Tech Conference on VLSI*, Cal Tech, 1979.

[4] G. Kissin. Functional Bounds on Switching Theory. In *Chapel Hill Conference on VLSI*, U.N.C., Chapel Hill, Computer Science Press, 1985.

[5] G. Kissin. Measuring Energy Consumption in VLSI Circuits: a Foundation. In *ACM Symposium on Theory of Computing*, ACM-SIGACT, 1982.

[6] T. Lengauer and Mehlhorn K. On the Complexity of VLSI Computations. In *Proceedings of CMU Conference on VLSI*, CMU, Computer Science Press, 1981.

[7] C. Mead and L. Conway. *Introduction to VLSI Systems.* Addison-Wesley, Reading, Mass., 1980.

[8] L. Snyder and A. Tyagi. *The Energy Complexity of Transitive Functions.* Technical Report TRCS-86-09-07, Dept. of Computer Science, University of Washington, Seattle, 1986.

[9] C.D. Thompson. Area-Time Complexity for VLSI. In *ACM Symposium on Theory of Computing*, ACM-SIGACT, 1979.

[10] J. Vuillemin. A Combinatorial Limit to the Computing Power of VLSI Circuits. In *IEEE Symposium on Foundations of Computer Science*, IEEE Computer Society, 1980.

Optimal 3-dimensional Layouts of Complete Binary Trees

Ajay K. Gupta and *Susanne E. Hambrusch*
Department of Computer Sciences, Purdue University
West Lafayette, IN 47907

A commonly used model for laying out VLSI circuits is to view the circuit as a bounded degree graph G in which the nodes correspond to processing elements and the edges correspond to wires. Graph G is then embedded in a two (2-d) or three (3-d) dimensional grid subject to layout constraints [6][4]. Within 3-d layouts, two models, the *One-Plane-Active* and the *All-Plane-Active model,* have recently received considerable attention [3][4]. In the first model only the grid intersection points on one of the boundary planes are allowed to contain nodes, while in the second model every grid intersection point can contain a node. In this paper we show how to optimally embed an n-node complete binary tree in the *All-Plane-Active* 3-d model when k, the size of one of the dimensions is given. For any given k, $1 \leq k \leq n/2$, we show how to obtain a layout of $\Theta(n + k \log n)$ volume.

The 2-d layout of complete binary trees has been studied extensively: If all the leaves of an n-node complete binary tree are required to be on the boundary, then $\Omega(n \log n)$ area is necessary and sufficient [6]. We refer to the layout placing all leaves on one of the longer sides of the layout as the *B-tree* layout. Furthermore, the *H-tree* layout achieves $\Theta(n)$ area by placing $\Theta(\sqrt{n})$ leaves on the boundary [6]. Optimal $\Theta(n + k \log n)$ area 2-d layouts have been described in [1]. Rosenberg describes $O(n)$ volume embeddings in 3-d grids for $k = n^{1/3}$ [4]. The 3-d results of our paper cover the entire range of k (i.e., $1 \leq k \leq n/2$). Our results can also be easily modified to get optimal 2-d layouts when k leaves of the tree, $(\sqrt{n} \leq k \leq n)$, are required to be positioned on the boundary. We refer to [2] for details.

Theorem 1. *Every n-node complete binary tree T can be embedded in a 3-d grid using $O(n + k \log n)$ volume, where one side of the grid is k and $1 \leq k \leq n/2$.*

Proof Sketch: First divide the tree T into $(k+1)$ subtrees $T_0, T_1, ..., T_k$. The subtrees $T_1, \ldots, T_k$ are the subtrees rooted at the nodes at level $\log k$ in T and T_0 is the remaining tree formed by levels 0 through $\log k$. Next consider a 3-d grid G_1 of dimensions $\sqrt{n/k} \times k \times \sqrt{n/k}$, which we view to consist of k planes, where each plane is of size $\sqrt{n/k} \times \sqrt{n/k}$. Place the H-tree layout of T_i on the i^{th} plane of G_1. Now move the root of T_i to the (say) top left corner of i^{th} plane. In order to find a layout of T_0, divide T_0 into l forests $F_1, \ldots, F_l$ $(l = \log k / \sqrt{n/k})$. Forest F_j consists of $\sqrt{n/k}$ levels of T_0, namely levels $(j-1)\sqrt{n/k}$ to $j\sqrt{n/k}$. Now consider a 3-d grid G_2 of size $\sqrt{n/k} \times k \times l$ consisting of l planes of size $\sqrt{n/k} \times k$ each. Place the B-tree layout of F_j on the j^{th} plane such that the leaves of F_j lie on the (say) left boundary if j is odd, and on the (say) right boundary if j is even. Next *merge* the roots in the forest F_j with the corresponding leaves in the forest F_{j+1} by routing wires from the j^{th} plane to the $(j+1)^{th}$ plane. By the *"merging"* of a leaf and a root we mean that a wire is routed from the leaf to the root and that the leaf node is deleted from the layout.

This work was supported by the Office of Naval Research under Contract N00014-84-K-0502 and by the National Science Foundation under Grant DMC-84-13496.

The final step is to combine G_1 and G_2 by laying grid G_2 on the top of G_1 such that the base dimension of G_2 matches the base dimension of G_1, the base dimension being $\sqrt{n/k} \times k$. Next, merge the root of T_i in G_1 with the i^{th} leaf of T_0 in G_2. The above construction gives the layout of T. The volume V of the final layout is:

$V = O(k * \sqrt{n/k} * (\sqrt{n/k} + \log k/\sqrt{n/k})) \leq O(n + k * \log n)$ ∎

Next we show that the layout given in Theorem 1 is optimal. Lower bound proof need only be given for k in the range of $(n/\log n, n/2]$.

Theorem 2. *In the All-Plane-Active model, any 3-dimensional layout of an n-node complete binary tree requires $\Omega(k \log n)$ volume, when one side of the 3-d grid is k and $n/\log n < k \leq n/2$.*

Proof Sketch: Assume that there exists a 3-d layout L_1 of the tree T using $o(k \log n)$ volume. Let the dimensions of L_1 be $l \times w \times k$ and w.l.o.g. assume $l \leq w$. We first transform the given $l \times w \times k$ 3-d layout into an $lk \times lw$ 2-d layout L_2 by projecting the 3-d grid onto a 2-d grid [3].

Let $n' = lk$ and let one of the longer sides of L_2 be s_0. We next show how to transform the layout L_2 into a layout L' of an $(2^{\lceil \log n' \rceil + 1} - 1)$-node tree T' that has all of its leaves positioned on one longer side of L'. The transformation will increase the area of L_2 by at most a constant factor. Depending on n' we distinguish two cases.

Case 1: $(n' > n/2)$: In this case we augment tree T by subtrees $T_1, T_2, \ldots, T_{n/2}$ of height $\lceil \log n' - \log n \rceil$ each. Every subtree T_i will have as root the i^{th} leaf t_i of T. This augmentation results in the tree T' of height $\lceil \log n' \rceil$. The layouts of trees $T_1, T_2, \ldots, T_{n/2}$ are added to the existing layout L_2 to get L' as follows.

- Pull the leaves t_i of T to the side s_0 of L_2 as described in [1]. Let m_i be the number of grid lines required such that distance between t_i and t_{i+1} is n'/n. Next introduce m_i new horizontal grid lines in L_2 for t_i below the horizontal grid line on which t_i lies. This gives layout L_2' of size $(3n' - n)/2 \times lw$ (worst case).
- Place the B-tree layouts of $T_1, \ldots, T_{n/2}$ in this order on a 2-d grid of size $(3n' - n)/2 \times \lceil \log n' - \log n \rceil$ such that the root r_i of T_i is positioned on the corresponding horizontal grid line on which t_i lies. This gives a layout L_3 in which the leaves of T_i's lie on the longer side s_2 and the r_i's lie on the opposite side s_1. Next join L_2' and L_3 at the side s_0 of L_2' and the side s_1 of L_3 and merge the t_i's and r_i's.

Case 2: $(n' \leq n/2)$: Prune the tree T below level $\lceil \log n' \rceil$ in L_2 and obtain the layout L' of area $o(n' \log n')$.

The new layout so obtained is the desired layout L' and has area $o(n' \log n')$, which is a contradiction since $\Omega(n' \log n')$ area is required. ∎

REFERENCES.

[1] P. Czerwinski and V. Ramachandran, *"Optimal VLSI Graph Embeddings in Variable Aspect Ratio Rectangles,"* To appear in *Algorithmica.* 1986.

[2] A. K. Gupta and S. E. Hambrusch, *"Optimal 3-dimensional Layouts of Complete Binary Trees,"* Technical Report CSD-TR-605, Purdue University, 1986.

[3] F. T. Leighton and A. L. Rosenberg, *"Three- Dimensional Circuit Layouts," SIAM Journal of Computing,* Vol 15, No 3, August 1986, pp. 793-813.

[4] A. L. Rosenberg, *"Three-Dimensional VLSI: A Case Study, " Journal of the Association of Computing Machinary,* Vol. 30, No. 3, July 1983, pp. 397-416.

[5] C. D. Thompson, *"A Complexity Theory for VLSI,"* Ph.D. Thesis, Carnegie-Mellon University, 1980.

[6] J. D. Ullman, *"Computational Aspects of VLSI,"* Computer Science Press, 1984.

An Efficient VLSI Architecture
with Application to Geometric Problems

(Preliminary Version)

V. K. Prasanna Kumar and Hussein M. Alnuweiri
Department of EE-Systems
University of Southern California
Los Angeles, CA 90089-0781

Abstract

In this paper we present a reduced hardware organization having n PEs and n^2 memory modules for solutions to several problems in computational geometry. This SIMD architecture has simple overall organization, low processing requirements and can be implemented using a limited VLSI chip set or off the shelf components. We implement fast parallel algorithms for several problems in computational geometry on planar as well as digitized points in the plane. We derive $O(n \log n)$ parallel algorithms for determining convex hull of n^2 planar points, geometric intersections, etc. With digitized input data, enumerating the extreme points of the convex hull, and related problems on a 0/1 image can be solved in $O(n)$ time; while many problems related to convexity of multiple objects can be solved in $O(n \log n)$ time. For several problems, the algorithms given are shown to be optimal. Finally, we compare the performance of the proposed architecture with an $n \times n$ 2-MCC.

I. INTRODUCTION

Parallel solutions to problems in computational geometry have recently been considered by several authors[MILL 84, PRAS 85]. For example, the 2-MCC of size $n \times n$ has been used to provide $O(n)$ solutions to many problems in computational geometry[MILL 84]. Even though these architectures provide substantial speedup for many computations, several practical considerations limit their implementation. Significant among these bottlenecks are: cost, I/O, matching problem size and array size. For example, the 2-MCC assumes an array with $n \times n$ PEs to a solve a problem on a image with $n \times n$ pixels. As the problem size increases it becomes impractical to build such arrays. Moreover, many problems in computational geometry have unique characteristics: low computation need, sparse communication between subcomputations and global information exchange. Thus, a large number of PEs remain inactive during parts of the computation, leading to inefficient use of the hardware.

In the current VLSI technology processing power still tends to be expensive compared to storage. Large capacity memory chips are offered by vendors while providing fast innovative access capabilities. In this paper, we explore an innovative architecture with reduced processing requirements for geometric problems that arise in Computer Graphics, and Image Processing.

The intent is to organize a large array of memory modules storing the data which is accessed by a small number of PEs so that linear speedup can be obtained. Such an organization can be implemented in the current VLSI technology using a limited chip set. The organization of memory, memory access, and data sharing leads to linear speed up over uniprocessor system for many problems. We illustrate the suitability of the architecture by showing efficient parallel algorithms for many geometric problems related to determining convexity and computing distances.

Using reduced processing power to solve a large size problem has been explored in several other contexts. In [JAJA 83] special purpose chips with small processing area have been designed to solve FFT, sorting, integer multiplication while in [TSEN 85] a general VLSI based multiprocessor organization is proposed to implement parallel algorithms for signal processing and numerical computations. In this paper, we present fast parallel solutions to problems in computational geometry. For planar point input we develop $O(n \log n)$ algorithms for determining the convex hull, estimating geometric intersections and other related problems. Problems such as finding a smallest enclosing box can be solved in $O(n \log^2 n)$ time. We provide $O(n)$ parallel algorithms for several problems including labeling figures, and convexity problems on digitized images. Many of the algorithms achieve optimum speedup compared to a uniprocessor system.

The rest of the paper is organized as follows: In the next section we describe a VLSI architecture. In section III we discuss parallel algorithms on the proposed architecture for planar point input. Section IV is devoted to efficient parallel algorithms for problems on digitized pictures. Concluding remarks, related architectures and comparisons to MCC are made in section V.

II. A VLSI BASED ARCHITECTURE AND DATA MOVEMENT OPERATIONS

In an $n \times n$ MCC there are n^2 PEs, each having a fixed number of registers and arithmetic/logic capability. The proposed architecture consists of n PEs which access a 2 dimensional memory with row and column access capability. The organization has n^2 memory modules with processor PE_i accessing the memory in the ith row and ith column. The PEs have arithmetic/logic capability and they can be general purpose microprocessors. The memory module M_{ij} corresponds to the local memory of PE_{ij} in the 2-MCC. The organization of such an architecture is shown in figure 1 for $n = 4$. Note that the memory module M_{ij} acts as a shared space between PE_i and PE_j. Each PE can also broadcast a single value to all memory modules in its row or its column. This architecture has the advantage that the processing area and the storage area are identified as two distinct components of the design. The entire memory can be designed as a single chip or a chip set with the current VLSI technology.

A memory row will be denoted by RM and a memory column will be denoted by CM. The jth memory module of the ith memory row (column) is denoted by $RM_i < j >$ ($CM_i < j >$). The simplified notation $RM < j >$ ($CM < j >$) will be used whenever we are carrying out an operation in which only the relative position of a memory location within a row (column) is important. The memory module M_{ij} stores the information in the local memory of PE_{ij} in the 2-MCC. The memory array can be accessed in two modes:

1. In this mode each PE can read or write from a single memory location in its row or column. A single bit is used to indicate whether the access is *row access* or *column access*.

2. In this mode each PE can only write a single value to all memory locations in its row or column. A single bit is used to indicate whether the access is row access or column access.

Each PE can also broadcast data to all other PEs in $O(1)$ time.

The reduced hardware architecture is configured as a SIMD (Single Instruction Multiple Data) computer. Each PE is basically an arithmetic/logic unit with a set of working registers and

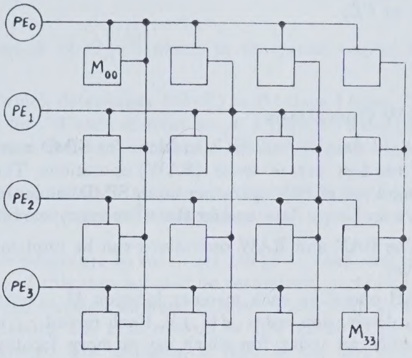


Fig. 1 : Reduced hardware architecture

flags. We now show how some standard data movement operations can be efficiently implemented on our architecture. Due to space limitations we will provide only a brief description of the operations. The details are given in the full version of this paper [PRAS 86].

Sorting

Many geometric algorithms require sorting as an initial step. An efficient sorting algorithm is, therefore, essential to the solutions to such geometric problems. In the following theorem we present an optimal sorting algorithm on the proposed architecture.

Theorem 1: Suppose we have n^2 elements stored one element per memory location. These elements can be sorted (into column major or row major order) in $O(n \log n)$ time on the proposed architecture.

Sorting can be done in $O(n (\log n)^2)$ time, by directly mapping the bitonic sorting algorithm for the 2-MCC onto our architecture. However, sorting can be done in $O(n \log n)$ time on our architecture, by using a combination of bitonic merge technique and sequential merge.

Tree Computations

The n PEs can be used to carry out tree computations on n entries such as computing the maximum, enumeration, or computing any other commutative, associative, binary operation. The following lemma states some of these results.

Lemma 1: Suppose that each of the n PEs contains at most one entry, then the following operations can be performed in $O(\log n)$ steps:

a. Computing the maximum of the entries.

b. Enumerating the entries. The enumeration order of an entry in PE_j, $0 \leq j \leq n-1$, is defined as the number of PEs containing an entry, say, to the left of PE_j.

RAR And RAW Operations

Two standard data movement operations for SIMD computers are the ***random access read*** (RAR) and the ***random access write*** (RAW) operations. These operations are used to move and copy data among a set of PEs operating in an SIMD mode. In our case, however, these operations are used to move and copy data among the n^2 memory modules using the n PEs.

Theorem 2: The RAR and RAW operations can be implemented in $O(n\log n)$ time on the proposed architecture.

In the RAR operation each memory location M_{ij} contains a record r_{ij} which consists of a data part and a destination index $d(i,j)$. Each record r_{ij} is to read the data in $M(d(i,j))$. A record may specify an index for which no memory location exists, in which case this record receives no data. Also a group of records may specify the same index $d(i,j)$. The idea of the algorithm which implements the RAR operation on our architecture is as follows: first sort the records, in row-major order, using the index $d(i,j)$ as the key. This brings each group of records with the same index in adjacent memory locations. From each such group of records choose a single record to be a ***representative*** of the group. Now perform the data distribution operation of lemma 2 on the representative records only, to send each such record to its destination. By letting each PE scan its RM, each representative record receives its data from the memory location containing it. Finally, sort back the records to bring each representative record immediately before its group of records. By letting each PE scan its RM, the data in each representative record is copied to all the records in its group. In the RAW operation, each record r_{ij} consists of a destination index $w(i,j)$ and a data part. Each record r_{ij} is to be written to $M(w(i,j))$. If many records specify the same index $w(i,j)$, then the record with the minimum data field is written into $M(w(i,j))$. (One can replace minimum with any other commutative, associative, binary operation.) RAW can be implemented using the same techniques used for RAR.

III. PARALLEL ALGORITHMS FOR POINTS IN THE PLANE

In this section we demonstrate efficient solutions to some fundamental problems in computational geometry on our architecture. We are concerned with sets of points (figures) comprised of $O(n^2)$ general planar points. The points are mapped onto the memory array such that each memory location M_{ij} contains a single planar point (x,y).

Our architecture can simulate the 2-MCC of size n^2 and the Hypercube of size n^2 with a factor of $O(n)$ loss in time. Since many problems can be solved on 2-MCC in $O(n)$ time, mapping such solutions directly onto our architecture leads to $O(n^2)$ time algorithms. However, we will present algorithms to solve many of these problems on our architecture in $O(n\log n)$ time. In many of our algorithms we use a combination of sequential and parallel methods in computational geometry. In such algorithms each PE performs sequential algorithms on the data in its RM or CM, and then the results are merged in parallel using all the PEs.

3.1 Convexity Algorithms

Constructing the convex hull of a set of points is essential to the solution of many geometric problems. In this section we present an $O(n \log n)$ parallel algorithm to construct the convex hull of a set S comprised of $O(n^2)$ points in the plane. Initially, these points are distributed one point per memory location. The convex hull of S, denoted by $CH(S)$, is defined to be the minimum convex set containing S. A point p is called a *vertex* of $CH(S)$ if p is not in $CH(S-p)$.

Theorem 3: The convex hull of a set S of $O(n^2)$ points in the plane can be constructed in $O(n \log n)$ steps on our architecture.

We present below an algorithm which determines $CH(S)$ in $O(nlogn)$ time. The basic idea of the algorithm is to find the convex hull of each disjoint set of n points in the plane, and then merge these hulls recursively to determine the convex hull of S [PREP 85].

1. Sort, using the sorting algorithm of theorem 1, the records in the entire memory array using x as the first key and y as the second key.

2. Now we use a recursive procedure to construct $CH(S)$. Each row contains n distinct points The procedure of this step is based on recursively partitioning the rows of the memory into two groups of equal number of rows. Each such group is called an *RM group*. The PE accessing the first row of an RM group is chosen to be the *leader* PE of that RM group. For example, if RM_i is the first row of some RM group, then PE_i will be the leader PE for that RM group. A flag F is set when a PE becomes a leader PE of some RM group. At any particular iteration each RM group will contain the representation of the convex hull of the points in its memory locations.

Initially, each RM group consists of a single RM. Each PE constructs the convex hull of the n points in its RM in $O(n \log n)$ time using any of the well known sequential convex hull algorithms [PREP 85]. Note that these convex hulls are disjoint. At each subsequent iteration we merge each pair of adjacent RM groups. This is done recursively by the leader PEs until all RM groups are merged into a single RM group which contains the final convex hull. By merging a pair of adjacent RM groups we mean constructing the joint convex hull which is the union of the two previous disjoint convex hulls contained in these two RM groups. The rows of the new RM group containing the new joint convex hull are just the union of the rows of the previous two RM groups. At each iteration the number of RM groups is reduced by half, while the number of rows of each new RM group (and thus the size of the convex hull it contains) is doubled. The final convex hull can thus be constructed in $O(log\ n)$ iterations. Each merge operation can be done in $O(log\ n)$ time on our architecture, as shown by the following lemma.

Lemma 3: Let $p_1, p_2, \cdots, p_i$ be the vertices of $CH(S_1)$ and $p_{i+1}, \ldots, p_{n^2}$ be the vertices of $CH(S_2)$, $0 \leq i \leq n^2$. Then the joint convex hull $CH(S_1 \bigcup S_2)$ can be constructed in $O(log(n^2))$ time.

In order to construct the convex hull of two disjoint convex hulls A and B, each has $O(n^2)$ points, we have to determine the left and right common tangents e_L and e_R of A and B, respectively. Using a binary search technique, these two tangents can be determined in $O(logn)$ time using a single processor [PREP 85]. Implementing this technique on our architecture is straight forward. Details are given in the full version of this paper. Note that in the above algorithm, step 1 takes $O(n)$ time while steps 2 and 3 take $O(n \log n)$ time.

Corollary: Given a set of n^2 labeled points in the plane, distributed one point per memory location, in $O(n \log n)$ steps the convex hull of each set of points with the same label can be constructed, for all the labels simultaneously.

3.2 A Smallest Enclosing Box

In this problem we are given a set S comprised of $O(n^2)$ planar points and we want to find a minimum area rectangle which encloses the set of points. Note that any such rectangle must enclose the convex hull of the set S. In [FREE 75] it was shown that such a rectangle has one side collinear with an edge of the convex hull, and each of the other three sides must pass through a vertex of the convex hull. A parallel solution to this problem is to find for each edge e of the convex hull a minimum area enclosing rectangle which has one side collinear with e. The rectangle with the smallest area among these rectangles is the smallest enclosing box for the set of planar points. In the following theorem we show how this technique can be efficiently implemented on the proposed architecture.

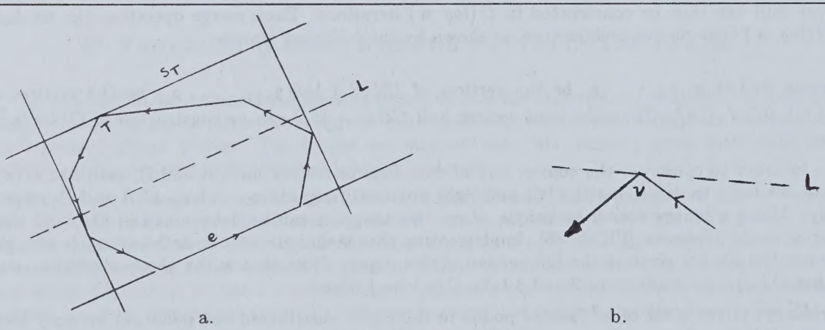
Theorem 4: Given a set S comprised of $O(n^2)$ planar points, distributed one point per memory location, a smallest enclosing box of the set of points can be determined in $O(n(logn)^2)$ time on the proposed architecture.

Suppose s_e is the side of the rectangle collinear with an edge e of $CH(S)$. We have to identify the other three sides s_T, s_L, and s_R of the rectangle as shown in figure 2a. We will only show how s_T is computed. The other two sides can be computed in a similar way.

The side s_T is computed by sweeping a line L parallel to s_e through the convex hull. Starting collinear with s_e, the line L is swept through the convex hull until it passes through a vertex T of $CH(S)$ which has the property that it is the last vertex of $CH(S)$ encountered as L is swept through $CH(S)$ (the sides s_L and s_R are computed in a similar way by sweeping a line vertical to s_e through the convex hull). The vertex T is called a *tangent* point. A vertex v of the convex hull is a tangent point if both of the edges incident on v lie on one side of the line L passing through v, as shown in figure 2b. Because of convexity it is easy to see that the vertex T can be identified in $O(logn)$ time using a binary search procedure. The following is the algorithm which computes a smallest enclosing box of S.

1. Construct $CH(S)$ as shown by Theorem 3. The vertices of the convex hull are distributed one per memory location and arranged in row major-order. The memory array also contains a list of edges arranged in row-major order.

2. For each memory location M_{ij} construct a record $Q_{ij} = < e_{ij}, L_{ij}, N_{ij}, D_{ij}, F_{ij} >$,



a. b.

Fig. 2

where e_{ij} denotes the convex hull edge in M_{ij}, L_{ij} denotes the line parallel to e_{ij}, N_{ij} is used by the binary search procedure explained below to specify the memory location containing the vertex to be inspected, D_{ij} denotes the binary search interval, and F_{ij} is a flag which indicates whether the next step is a forward search or a backward search (according to the row-major ordering). Initially, N_{ij} points to the M_{00} for $0 \leq i, j \leq n-1$, and $D_{ij} = s/2$, where s is the number of convex hull vertices.

3. **Global binary search procedure:** Let $v[M(p,q)]$ denote the convex hull vertex in memory location M_{pq}. The following procedure determines the side s_T for each edge of the convex hull:

   i. Each PE adds the contents of D_{ij} to N_{ij} for each record in its RM. Now N_{ij} of each record contains the address of the memory location containing the vertex to be inspected.

   ii. Perform RAR so that each record Q_{ij} reads the vertex $v[M(N_{ij})]$.

   iii. Each PE now scans its RM and determines for each record Q_{ij} in its RM, if L_{ij} is tangent to $v[M(N_{ij})]$ or not. If it is, then we are done and the side s_T is determined. Otherwise, set the flag F_{ij} to forward or backward according to the location of intersection, divide the interval in D_{ij} by 2, and go back to step i.

   The above binary procedure determines the side s_T for each edge e of $CH(S)$ in $logn$ iterations. The other two side, S_R and s_L, can be determined in a similar way.

4. Now each memory location contains the four sides of a rectangle containing the convex hull. The rectangle with the minimum area can be identified in $O(n)$ time.

Step 1 of the algorithm takes $O(n \log n)$ time, while steps 2 and 4 take $O(n)$ time. The binary search procedure of step 3 consists of $\log n$ iterations in the worst case. Each iteration takes $O(nlogn + n)$ time, and thus the total time is $O((n \log n + n) logn)$ which is still $O(n (logn)^2)$.

IV. RELATED APPLICATIONS TO DIGITIZED PICTURES

When black and white images of objects are processed by a computer, they are often represented by a two-dimensional rectangular array of 0s and 1s. In an $n \times n$ digitized image the location of each point (pixel) is specified by the index (i,j), where i and j are integers which specify, respectively, the row number and the column number of the point. Many algorithms related to a single digitized figure can be implemented on the proposed VLSI array in $O(n)$ time, the same as the time taken by the 2-MCC with n^2 processors to implement these algorithms. $O(nlogn)$ time is required to implement most of the algorithms related to multiple digitized figures. As in section III, many of these algorithms use a combination of sequential and parallel methods in computational geometry. Moreover, given the convex hull of an $n \times n$ digitized figure, many properties such as computing the diameter and a smallest enclosing box can be computed in $O(logn)$ additional time [PRAS 86]. This is possible because of the broadcasting feature, and the ability of each PE to perform binary search on the items in its RM (or CM). Broadcasting removes the $O(n)$ bound needed to copy the items in one memory row to all other memory rows, while binary search enables each PE_i to compute a property with respect to the ith item in its RM in $O(logn)$ time. In the following discussion it is assumed that the points of the digitized picture are mapped onto the memory array such that the point (i,j) is stored in memory location M_{ij}. Next we show how the proposed architecture provides an efficient solution to the labeling problem. Other applications can be found in [PRAS 86].

Labeling connected regions of 1s

An $n \times n$ image may contain more than one region of connected 1s. Fundamental to many problems in image processing, is to identify figures which are connected regions of 1s in the image. The labeling problem is to identify a unique ID with each figure in the image. The ID associated with a figure for example, can be the index of the lower leftmost pixel in that figure. An elegant approach is to use a recursive procedure in which blocks of size $k \times k$ are labeled and using the adjacency information along the boundary of the blocks, blocks of size $2k \times 2k$ is labeled. This procedure results in $O(n)$ time solution to the above problem on a 2-MCC of size n^2 [PRAS 86a]. This problem can also be solved in $O(n)$ time on the reduced VLSI array.

Theorem 5: The labeling problem for an $n \times n$ 0/1 image can be solved in $O(n)$ time on the proposed architecture.

The idea is to label figures within blocks of size $\frac{k}{2} \times \frac{k}{2}$ and paste together information in four adjacent quadrants to label the figures in a block of size $k \times k$ and repeat this procedure. Notice that in order to paste the four adjacent blocks, the connectivity information is contained in the two inner boundaries of length k as shown in figure 3. Consider for example, the merging of blocks 1 and 2. At this time the pixels within each of the the blocks are labeled with unique ID of the figure within the block. In order to do the merging, we represent each figure in blocks 1 and 2 which intersect the boundary by a node and the connectivity information along the boundary determines the edges incident on these nodes. Thus, if vertices v_1 and v_2 represent figures with labels l_1 and l_2 an edge exists between v_1 and v_2 iff there exists (i,j) such that the pixel at (i,j) is labeled l_1 and the pixel at $(i,j+1)$ is labeled l_2 and pixels (i,j) and $(i,j+1)$ belong to blocks 1 and 2 respectively. The resulting graph is a undirected bipartite graph with at most $2k$ vertices and at most $2k$ edges. The connected components in the graph can be found in $O(k)$ time using a uniprocessor [AHU 79]. Repeating this along the horizontal boundary, the ID of the figures in the quadrants 1,2,3,4 can be found in $O(k)$ steps. In order to implement this idea on the proposed array, we divide the $n \times n$ memory into disjoint regions of size $n \times k$ as shown in figure 3. We call each such region a *column strip*. The k PE's in a column strip operate in

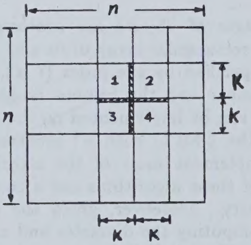


Fig. 3

parallel to perform the merges in the column strips. As the blocks are merged we only update the labels of the memory locations along the outer boundary s shown in figure 3.

A digitized image may contain more than one figure. In this case, the points of each figure are labeled with a single distinct label. Since all the figures are in an $n \times n$ mesh, they can be represented by an $n \times n$ matrix $P(i,j)$ whose entries are figure labels or 0s. The matrix P is mapped onto the architecture so that M_{ij} contains the label $P(i,j)$, $0 \leq i,j \leq n-1$.

Theorem 6: Given an $n \times n$ image in which all the figures have been labeled, one can construct the convex hull of each figure, for all figures simultaneously in $O(n \log n)$ time.

V. CONCLUSION

Geometric problems have an increasing importance in many areas such as Computer Graphics, Robot Vision, VLSI layout design, etc. Parallel algorithms have been proposed to solve many of these problems. Such parallel algorithms have been typically implemented on the 2-MCC and Pyramid Computer. Such array computers consist of $O(n^2)$ processors to solve problems on n^2 points. Our research is motivated by the fact that a large number of processors is not active during the execution of many parallel geometric algorithms on the 2-MCC. The architecture we have presented uses only n processors and n^2 memory modules to solve problems on figures with n^2 points. We have shown that many parallel geometric algorithms can be implemented efficiently on the proposed architecture.

The organization of the PEs and the memory modules has many significant features. In many algorithms in which the data in a RM or a CM is sorted according to a certain order, the PE accessing this RM or CM can identify an entry in $O(\log n)$ time using binary search. For example, given the representation of the convex hull of a digitized figure, the diameter and a smallest enclosing box can be determined in $O(\log n)$ extra time. However, such an operation takes $O(n)$ time on the 2-MCC. Another feature of our architecture is that it utilizes sequential as well as parallel techniques in implementing many algorithms. Each PE can perform well known sequential algorithms on the data in its RM or CM, and then these results can be merged in parallel using all PEs. Thus we can make use of the available sequential techniques as an initial step in implementing algorithms on our architecture, and concentrate our efforts on designing efficient parallel merging techniques to combine the results.

Table 1 compares the running time for many geometric algorithms on the 2-MCC and our architecture. These results show that all the algorithms presented in this paper run on our architecture in a number of steps comparable to the 2-MCC, especially for the case of digitized images.

TABLE 1

Time for computing some geometric properties (in order notation)

	2-MCC	Reduced Hardware Architecture
Number of PEs	n^2	n
Planar points input		
Convex hull algorithm		
a. Single figure	n	$n\log n$
b. Multiple figures	n	$n\log n$
Smallest enclosing box	n	$n \log^2 n$
Digitized pictures		
Convex hull algorithm		
a. Single figure	n	n
b. Multiple figures	n	$n \log n$
Labeling figures	n	n

VI. SELECTED REFERENCES

[AHU 79] A. Aho, J. Hopcroft, J. Ullman, " Design and analysis of computer algorithms", Addison Wesley, 1979.

[FREE 75] H. Freeman and R. Shapira, "Determining the minimal-area encasing rectangle for an arbitrary closed curve", Communications o the ACM, vol. 18, 1975.

[JAJA 83] Joseph Ja'Ja' and Robert M. Owens, "An architecture for a VLSI FFT processor", INTEGRATION the VLSI journal 1, 1983.

[MILL 84] R. Miller and Q. F. Stout, "Computational geometry on a Mesh-Connected Computer", proc. 1984 International Conference on Parallel Processing.

[PRAS 85] V. K. Prasanna Kumar and C. S. Raghavendra, "A Enhanced mesh connected VLSI architecture for Image Processing", proc. International Conference on Computer Vision and Pattern Recognition, 1985.

[PRAS 86] V. K. Prasanna Kumar and Hussein M. Alnuweiri, "An efficient VLSI architecture with applications to geometric problems", Tech. Report, USC, 1986.

[PREP 85] F. Preparata and M. I. Shamos, Computational Geometry: An Introduction. Springer-Verlag, New York, 1985.

[SHAM 75] M. I. Shamos, "Geometric Complexity", proc. Annual Symposium on Theory of Computing, 1975.

[TSEN 85] P. S. Tseng, K. Hwang and V. K. Prasanna Kumar, "A VLSI based multiprocessor architecture for implementing parallel algorithms", proc. 1985 International Conference on Parallel Processing.

VLSI SUPERCOMPUTING

SWAMINATHAN MANOHAR and **GERARD BAUDET**

Brown University, Providence, RI.

Abstract: VLSI systems dedicated to solving a very narrow set of problems and capable of being attached to a wide range of hosts, are proposed as a means of low-cost supercomputing. Such systems are termed super-coprocessors. The advantages of this approach to supercomputing will be illustrated using the architectures of Poisson Engine-I (PE-I) and its successor PE-II, super-coprocessors for solving partial differential equations.

1. Introduction

The increasingly large number of application areas which require supercomputer performance has spurred research into supercomputing. Many different approaches ranging from tightly coupled multiprocessors to massively parallel architectures are being pursued. We propose a new methodology for supercomputing which utilizes the potential of VLSI to provide highly parallel systems with hundreds of processing elements.

The mainstream research in the next generation supercomputers can be exemplified by projects like the NYU Ultracomputer [1], RP3 of IBM [2], and the Cosmic Cube at CalTech [3]. All these machines have an underlining similarity: a large number of fairly autonomous processors interconnected to cooperate on the execution of a wide range of problems. Since these supercomputers are intended to be fairly general purpose machines the following complexities are associated with them:

- The individual Processing Elements with local storage and control are fairly complex.
- A sophisticated interconnection network is needed to accommodate various communication geometries.
- The systems and application software needed to efficiently utilize the large number of processors in solving a range of problems can be expected to be very complex.

In short, such machines magnify the complexity and the bottleneck associated with conventional von Neuman machines a thousand-fold. Even if solutions could be found for the above problems, the cost of such systems will make them inaccessible to a large percentage of the research community.

We take a different approach based on the following :

VLSI: The advent of Very Large Scale Integration provides extremely large amount of computational hardware at very low cost. This effectively gives unlimited processing power to the computer architect. The architectures of the supercomputer projects mentioned above have been designed based mainly on conventional multiprocessor system concepts. Designers of these systems depend on VLSI to provide feasible implementations. Hence VLSI is being considered just a tool to implement proposed architectures. We maintain that to realize high performance systems VLSI should have a more direct influence on the architecture. suitable for VLSI implementation.

Systolic Architecture: This is an architecture highly suitable for VLSI implementation. The term systolic architecture refers to the methodology of mapping high level computations into hardware structures, developed at CMU. The chief characteristics of such structures are summarized below:

- Network of a few types of simple processing cells.
- Regular geometry of the cells and their interconnection.
- Localized communication.
- Modular and hence extensible implementation.
- High throughput without associated increase in memory bandwidth.
- Extensive hardware concurrency.

(For more details about systolic architecture the reader is referred to [4,5]).

Coprocessors: The concept of a coprocessor can be illustrated with the example of Intel's 16-bit microprocessor 8086 and the numeric coprocessor 8087. The 8087 has a complete set of high precision floating point arithmetic and connects one-to-one with the 8086. During the execution of an application program the required floating point computations are done in parallel by the 8087. The application program need not be rewritten to run on a system without the 8087 and can be executed using an emulator for the 8087. The basic idea behind the concept of a coprocessor is the use of special purpose hardware to decrease computation time and to increase concurrency without the specific intervention of the application programmer.

1.1. Supercomputing with VLSI

We define the term Super-CoProcessor (SCP) to refer to a VLSI system tuned to solving a particular computation intensive problem. The SCPs incorporate the design paradigms discussed above to provide highly efficient computing structures. Such SCPs could be connected to a variety of host machines and function as very high speed hardware library calls. The design goal of an SCP is to provide supercomputer performance for a single problem class at a few orders of magnitude lesser cost than current day supercomputers. The performance/cost advantage of SCPs stem from two related factors:

- The dedicated hardware of the SCPs can be tailored to fit the algorithm structure resulting in very high performance.
- They are special purpose machines and hence the overhead in terms of system and application software is considerably less than general purpose machines.

A number of problems commonly encountered in large-scale scientific computation could benefit from the use of SCPs. Solution of Navier-Stokes equations in fluid dynamics, matrix computations and Fourier transforms in signal processing, hidden-surface elimination and rendering in real-time graphics, and Monte Carlo simulations are a few examples.

We propose a collection of SCPs controlled by a general purpose host machine as a cost effective supercomputing alternative. It is very rare that a given research community requires supercomputer performance for a wide range of problems. More realistically they are interested in a small set of problems. Their needs could be cost effectively met by a few SCPs rather than a current day supercomputer.

In this paper we demonstrate the feasibility and the cost/performance advantages of SCPs by describing the architecture of a super-coprocessor for solving Partial Differential Equations (PDEs). The rest of the paper is organized as follows: In the next section the PDEs to be solved and the method of solution are outlined. In section 3 the architectural details of Poisson Engine-I (PE-I) and its more general and powerful successor PE-II are described. An estimate of the performance of these machines and the status of the research are presented in the last section.

2. Problem description

We are interested in the solution of second order linear partial differential equations in two dimensions, of the form,

$$a\frac{\delta^2\psi}{\delta^2 x} + b\frac{\delta^2\psi}{\delta^2 y} + c\frac{\delta x}{\delta x} + d\frac{\delta\psi}{\delta y} + e\psi + f = 0 \qquad \text{--(1)}$$

where, a to e can be constants or functions of x and y and f can be a constant or function of x, y and ψ.

Two special cases of this equation are the Poisson and Laplace equations. Poisson equation figures prominently in the solution of Navier Stokes equations and can be written as follows:

$$\frac{\delta^2\psi}{\delta^2 x} + \frac{\delta^2\psi}{\delta^2 y} = f(\psi)$$

The Laplace equation is a special case of the Poisson equation with the right hand side being zero. In this paper we will describe in some detail, Poisson Engine-I (PE-I) which is designed to solve the Laplace equation. The logical design of PE-I is complete and the design of three custom VLSI chips for the prototype implementation is in progress. We will also discuss research in progress to generalize the architecture of PE-I to arrive at its more powerful successor Poisson Engine-II which can solve equation (1). One of the most popular ways of solving such equations is the finite difference method and is briefly described below.

The problem: Given the boundary conditions over a rectangular area, find the value of the function u at all points within the boundary subject to the Laplace's equation,

$$\frac{\delta^2\psi}{\delta^2 x} + \frac{\delta^2\psi}{\delta^2 y} = 0$$

The finite difference method imposes a rectangular grid over the area of interest and approximates the values at all points within a square region surrounding a grid point by the value at the grid point. In effect the problem is reduced to finding the function value at these grid points. The finer the grid the closer will be the approximation. Typically, these values are computed by an iterative procedure. At each step of the iteration the value at a grid point is replaced by the average of its four neighbors. If the governing equation is (1) instead of Laplace's equation, then at each step the computation will be a weighted average of its neighbor values. For details of the method and its derivation the reader is referred to any text on numerical solution of PDEs.

3. Architecture and design details of Poisson Engine

The finite difference method has been chosen in our solution since it exhibits a number of properties desirable from the perspective of VLSI implementation:

- There is potential for large concurrency.
- The basic computation step at each grid point is very simple.
- The computation at each grid point requires data from its nearest neighbors only.

It is clear that if one Processing element (Pe) per grid point is available over the entire grid, high concurrency could be achieved. The Pe at each of the grid points has to successively find the average of its four neighbors until some error criteria are met. Such an architecture with a mesh of processing elements with near neighbor communication was first suggested by [6]. For such an architecture to become practical two important questions have to be answered. The first relates to the finite size of the hardware. Any physical system which implements the above architecture can have only a finite number of Pe's. The first question then, is how to utilize a finite sized array of Pe's to solve a much larger problem? The second related question is, how to allocate the available Pe's in the most efficient manner?

A number of proposed architectures for this problem have either ignored the first question by assuming that the size of the processor array corresponds to the problem size (see [7] for example), or proposed a solution by providing RAM for each Pe ([8,9]). The latter approach increases the complexity of individual Pe's and makes the architecture difficult to scale up.

We solve the first problem by using the results from asynchronous iteration techniques and the systolic design paradigm. We answer the second question later in this paper. Systolic architectures have been introduced in Section 1. A brief description of asynchronous iteration techniques is given below.

3.1. Asynchronous iteration

Asynchronous iteration techniques [10], are the generalization of *chaotic relaxation* methods developed by [11]. The main result of interest is that, irrespective of the order in which iterations are performed or the relative number of iterations performed at the various grid points, classical iterative methods, such as Jacobi, Gauss-Seidel, and SOR iterations will converge. That is, it is not necessary to perform one iteration (averaging operation) at every grid point in the problem domain before starting the next. Iterations could be performed in any order using any available previous values subject to some very loose restrictions and still the iterations will converge to their proper values. It must be noted that though in the sequential domain asynchronous iteration methods are much slower than a number of other algorithms, they have the following desirable properties:

- Very highly parallelizable (it is possible to attain a speedup linear in the size of the hardware)

- Provide a basic framework for the solution of PDEs using much faster algorithms like the Multilevel Adaptive Grid techniques.

The use of asynchronous iteration techniques and the systolic design methodology results in a high performance architecture and is described in the next section.

3.2. Architecture of the PE-I

The block schematic of the PE-I is shown in Figure 1. The basic computation at a grid point in the finite difference method is to find the average of the values at the nearest neighbors. This computation is performed by a simple *Processing element* (Pe). An array of Pe's called the *Operator* is globally controlled by the *Operator Control Unit* (OCU). The OCU broadcasts instructions to the Operator through the control bus and transfers data through the data bus. The OCU communicates with the host to get the problem description and the data. The data is fed in a systolic fashion to the Operator and is used a number of times before being returned to memory. Hence high throughput can be achieved even with a moderate communication bandwidth between the OCU and the host. The block named *Interface element* (Ie) in Figure 1 performs the following functions:

- Provides an interface between the Operator and the OCU to transfer data in both directions,
- Simulate the actions of the missing neighbors for the Pe's at the boundary of the Operator,
- Perform relative error computations.

Operation of the PE-I: The design aspects of the components of PE-I could be better understood if the overall operation is explained first.

Initialization phase: The host initiates the computation by sending the problem information (problem size, error criteria, etc.) to the OCU. The OCU loads the Operator with the initial values and the Interface elements with the error threshold.

The computation phase: It has to be kept in mind that in general the dimensions of the Operator will be much smaller than the dimensions of the problem domain. Hence the Operator has to perform the averaging operations one block of grid points (of size equal to the Operator size) at a time. It is helpful to view the flow of data from various grid points into the Operator as the 'movement' of the Operator over the corresponding grid points in the problem domain. With this view in mind we define two terms, the Operator step and the Operator cycle. An Operator step is the averaging operation performed by all the Pe's of the Operator at one position in the problem domain. An Operator cycle consists of all the Operator steps as the Operator scans the domain as shown in Figure 2. After reaching the top left corner the Operator retraces its steps back to the starting position. This is one of a number of methods of moving the Operator over the problem domain to ensure a smooth data flow through the Operator. The Operator cycles are repeated until it is determined by the OCU that the iterations have converged. At this point the computation phase is complete.

Termination: The OCU informs the host about the completion of computation.
The next section describes the design details of the components of PE-I.

3.3. Design details

The initial implementation of PE-I has the following features:

- An Operator element (Oe) with a 4x4 array of Processing elements (Pe's).

- An Interface element (Ie) realized by means of two chips, the Serial Chip (SC) and the Parallel Chip (PC).

- A commercial microprocessor (MC68000) as the OCU

- 32-bit integer arithmetic

- A scaling facility to enable the OCU to implement a limited floating point capability.

- Bit-serial communication between any two Pe's and between the Oe and the Ie.
- Parallel communication between the OCU and the Ie.
- Simple arithmetic capability for the Pe's: each Pe can find the average of its four neighbors and move the result to any of the four neighbors.

We discuss briefly, the reasons for some of the design decisions, the limitations imposed on the system because of these decisions and the modifications incorporated in PE-II to overcome these limitations.

The major objective of the design has been to make the hardware simple and very modular. Thus, only three kinds of chips are required to build an array of any size and its interface to the OCU without the need for any 'glue' chips. It is obvious that floating point arithmetic will be mandatory in any practical system, but it was felt that the added complexity due to its implementation in a prototype system will not be commensurate with any architectural insights it might provide. To alleviate the problems due to this limitation the scaling facility has been provided.

Scaling enables the OCU to simulate a floating point computation by holding an exponent common to all the Pe's in the Operator. This assumes that all the points that fit in the Operator have only a 'reasonable' range of values. As the computation proceeds, the OCU determines if the values in the Operator are close to either overflowing or underflowing the 32-bit mantissa and suitably rescales all the Pe values. The exponent stored in the OCU is adjusted to reflect the new scale factor. A control line provided in the Operator enables the OCU to scale up or down, the values of all the Pe's within a few clock cycles.

Pin limitations, the simplicity of the computation to be performed by the Pe and the requirement of modularity contributed to the decision to make communication between Pe's bit-serial. Potentially, this might cause reduced performance of the PE-I, but as will be seen later, due to overlapping communication with computation the effect of bit-serial communication has been considerably mitigated.

The simplicity of the Pe is in line with the philosophy behind SuperCoprocessors: a large array of Processing elements just sufficiently complex to solve the given problem efficiently. The Pe's do not have any program memory or data memory other than a couple of registers and are adequate to solve Laplace equations.

In spite of these apparent limitations, due to the concurrency and the efficient data-flow, as shown in Section 3.4, PE-I provides considerable throughput, bringing out the merits of the SCP paradigm.

Enhancements in PE-II: A number of orthogonal extensions can be incorporated to make PE-I more powerful:

- Provide increased arithmetic functionality to the Pe.

This will mean the ability to compute weighted averages rather than a simple average of four neighbors as in PE-I. Hence more general equations can be solved using more efficient solution methods like SOR (Successive Over Relaxation) techniques.

- Provide floating point capability to the Pe's.
- Use multiplexed bit-parallel communication between Pe's and between the Operator and Interface element.

This is not completely orthogonal to introducing floating point capability, since at the current levels of integration at most one Pe with floating point arithmetic can fit on a single chip. With just one powerful Pe per chip, the pin limitation is not severe and it becomes more meaningful to have efficient parallel communication.

The architecture of PE-II will incorporate a combination of these extensions. Other than these modifications, the overall architecture, the flow of data and control from the OCU remain the same as that of PE-I, thus providing considerably enhanced performance. In the next section, some of the functional and design details of the components of PE-I are presented.

3.3.1. The Operator Control Unit

The functions of the OCU are the following:

1) Communicate with the host to get the problem details

2) Manage the PE-I memory,

3) Maintain a continuous stream of data to and from the Operator,

4) Generate and broadcast the control instructions necessary for the proper functioning of the Operator, and

5) Determine when to terminate the computation based on the information provided by the Interface elements.

The instruction repertoire of the OCU of PE-I is very simple. There are instructions to load the Operator with the data values, the Interface element with the error threshold values, computing the average at each Pe and moving the result to any of the four neighbors.

The PE-I memory organization is a major performance determining factor. Problems involving solution of PDEs generally have very large memory requirements. A complete memory hierarchy, except for the disks, will be directly controlled by the OCU. The OCU will depend on the memory management system of the host to access the disks. management system of the host accessing the pages required by the OCU will be used. The very regular data-flow requirements of the computation implies that a very efficient prepaging scheme could be used.

The control program for the OCU is the only piece of software required by PE-I. Programming the OCU is a flexible way of experimenting with various alternatives on how to employ the Operator most efficiently. Typically after the research stage this program rarely needs to be rewritten, and the application programmer on the host can be unaware of even the existence of such a program and consider the PE-I to be a hardware library. In PE-II the instruction set of the OCU has to be more complex, to reflect the added complexity of the Pe's. The coefficients in equation (1) can be functions of the spatial coordinates and so the OCU has to give a sequence of instructions to evaluate these functions, simultaneously at all the Pe's. Instructions are also needed to manipulate the stack of registers provided in the Pe's.

3.3.2. The Operator and Interface elements

Figure 3 shows a sample Pe with its neighbors. There are two distinct aspects to the function of each Pe, the actions performed during each time step and when the Operator moves one step. During each time step,

1) Add the values of its four neighbors and take their average,

2) Send its current value to its four neighbors.

When the Operator moves one step (say, to the right),

3) Send its current value to its left neighbor,

4) Get the current value of its right neighbor.

The entire computation proceeds synchronously. Hence there is no need to provide buffers between Pe's . The current value from a Pe could be directly routed to the adders of its four neighbors. To speed up the addition of four neighbor values, each Pe adds up all the four bit-serial numbers simultaneously. The output of this adder goes to an accumulator. The division by four is accomplished by shifting the accumulator. After the averaging operation, the contents of the accumulator are transferred in parallel to a current-value register. During the addition process, the current value of the Pe is sent to its four neighbors bit-serially by shifting out the contents of the sum register.

The Interface element is realized by means of two chips: the Serial Chip (SC) and the Parallel Chip (PC). Their major functions are the same as what their names imply. The PC has a parallel interface to the OCU and performs the parallel to serial conversion. The converted data is communicated to the SC. Loading of the PC and the parallel to serial conversion are pipelined with the computation performed by the Operator. The internals of the PC are quite simple. There is a stack of parallel-in-parallel-out and serial-in-serial-out registers. The SC and PC are controlled by the OCU using the control bus. The Serial Chip connects bit serially to the Operator on one side and the PC on the other. It measures the error between successive iteration values in the Pe's and compares with the error threshold.

The Interface Element in PE-II will be modified according to the extensions over PE-II. For example, if there is bit-parallel communication, then the Ie will consist of just the PC with the error computation circuitry added. If just the floating point capability is added, the only modification will be to change the error computation in the SC to be implemented in floating point.

3.4. Performance estimate

A number of conservative assumptions have been made in arriving at an estimate of the throughput possible with the PE-I. Let the dimensions of the Operator element (which is realized as a single chip) be $a x a$. Since the Interface element is designed to fit an Oe, it will have a dimension of a elements. Let the dimensions of the Operator built using a number of Oe's be LxH. Let t be the time for one averaging operation. The frequency of the control clock of the Operator is assumed as 10 MHz. The word length of integers to be used in PE-I is 32 bits.

Number of clock cycles required for the averaging operation	= 34
Clock cycles required for moving the Operator one step	= 32

Hence assuming that one move operation is performed after every averaging operation, time for one Operator step $t = 66 \times 0.1 \mu\text{sec} \approx 7 \mu\text{sec}$

The entire Operator computes LxH averaging operations (four arithmetic operations each) every t microseconds resulting in a throughput of $(4/t)(L \times H)$ MOPS (Million arithmetic Operations Per Second). The following gives the number of Operator elements (with $a x a$ Pe's each) and Interface elements required for building the LxH Operator.

Number of Oe's	=	$(L \times H)/a^2$
Number of Ie's	=	$2\,(L + H)/a$
Total chip count	=	$4\,(L + H)/a + (L \times H)/a^2$

The required I/O bandwidth between the OCU and the Operator depends on the direction of movement of the Operator. Consider the Operator moving in its lengthwise direction. At each Operator step, H new words have to be loaded into the Ie's and H words read from the Ie's, requiring a I/O bandwidth of 2H/t Mega words per second. Similarly when it moves in a direction perpendicular to its length, the bandwidth between the OCU and the Operator has to be 2L/t Mega words per second. Table 1 lists some of these measures for various Operator sizes and shapes. An Operator element with 4x4 Pe's has been assumed. Sixteen Pe's could comfortably be laid out to fit in a 48-pin package.

A number of observations about Table 1 are in order. First, it is seen that with less than two hundred reasonably sized chips a throughput of over 500 MOPS can be achieved. Second, the shape of the Operator has a considerable influence on the chip count and the I/O bandwidth between the OCU and the Operator. This and a number of other factors which influence the system performance will be discussed next.

4. Performance enhancement

In [12] Kung proposes a 'Trillion Operation machine' which utilizes speedups due to the following factors:

> "Since I/O operations, rather than processors, are expected to be the bottleneck, we consider here only factors that are able to either increase I/O bandwidth or decrease I/O requirements.
>
> - A factor of 20 increase in I/O bandwidth from memory chip speed improvements.
> - A factor of 100 increase in the I/O bandwidth by using interleaved memory, multiport memories, and high-bandwidth interconnection networks.
> - A factor of 100 decrease in I/O requirement by using systolic arrays of 100-cell depth, i.e., only one I/O needed for every 100 operations.
> - A factor of 5 decrease in the I/O requirement by allowing each cell of the systolic array to perform several arithmetic operations in one cycle."

In addition to these, since PE-I is a special purpose machine a number of problem dependent schemes for enhancing performance can be applied. These include techniques for improving the rate of convergence of the iterations, assigning more computation time to parts of the problem domain with very low rate of convergence and the use of efficient algorithms like the multigrid methods. The rest of the section discusses these possibilities in more detail.

4.1. Breaking the I/O bottleneck

Of the four ways of improving I/O performance mentioned by Kung [12], PE-I employs the last two. Each Pe in the Operator corresponds to a cell in a two-level pipelined systolic array [13]. The Operator step corresponds to the major systolic cycle and the system clock corresponds to the second level pipelining clock. Hence at each Operator step a number of arithmetic operations are performed (addition of four numbers and the division of the sum by 4). The decrease in I/O requirement due to the number of Pe's in the Operator depends crucially on the shape of the Operator. For example, the Operator with dimensions 8x128 requires an I/O bandwidth of 2.28 million words/sec when moving along its lengthwise direction. This is considerably less than the bandwidth required by the Operator with dimension 32x32 (9.14 million words/sec with the same total number of Pe's). The 8x128 Operator, however, requires more bandwidth when it moves in a direction perpendicular to its length (36.58 million words/sec), and this movement occurs (with reference to Figure 2), whenever the Operator reaches the boundary.

With an available bandwidth of 2.28 Mwords/sec for the 8x128 Operator there are two options to use: 1) idle the Operator till all the 128 words of data are loaded or, 2) let the Operator perform more than a single iteration in the same position until all the data is loaded. The first option will reduce the maximum throughput but the average will still be higher than that of a 32x32 Operator for a given I/O bandwidth. The second option appears to be more attractive since in addition to maintaining a higher throughput, initial simulation results indicate that multiple iterations at each Operator step decrease the total I/O requirements.

The control program of the OCU can determine the areas of low convergence and can employ the Operator to perform more iterations in those areas. This can decrease the total number of Operator cycles required. Another software enhancement will be to use multi-grid methods for solving PDEs [14,15]. These methods use a sequence of finer grids to complete the computation with considerably less number of iterations.

5. Conclusion

Super-coprocessors like the PE-I have been presented as a feasible low cost supercomputing alternative. The architecture and design details of PE-I, a system for solving Laplace equations have been described and an outline of possible enhancements to realize its successor PE-II has been given. It has been shown that throughputs of the order of 500 MOPS are possible with processor arrays built using less than 200 chips. The implementation of the three custom chips required for the system is in progress and a prototype system is expected to be completed by June 1987. Research is in progress to bring other problems like sorting into the Supercoprocessor framework.

References

1. A. Gottlieb et al., "The NYU Ultracomputer--Designing an MIMD shared memory parallel computer," *IEEE Trans. on Computers* **C-32**(2) pp. 175-189 (February 1985).
2. G. F. Pfister et al., *The IBM Resarch Parallel Processor Prototype (RP3): Introduction and architecture.* August 1985.
3. C. Seitz, "The Cosmic Cube," *Communications of the ACM* **28**(1) pp. 22-33 (January 1985).
4. H. T. Kung, "Why Systolic Architectures," *Computer* **15**(1) pp. 37-46 (Jan 1982).
5. S. Manohar, "Systolic architectures: A critical survey," Tech. Report, Brown University (1986).
6. E. C. Gritton et al., "Feasibility of a special-purpose computer to solve the Navier-Stokes equations," R-2183-RC, RAND corporation (June 1976).

7. S. Y. Kung et al., "Wavefront Array Processor: language, architecture, and applications," *IEEE Trans. on Computers* **C-31**(11)(1982).

8. W. Handler et al., *DIRMU multiprocessor configurations.* August 1985.

9. T. Hoshino et al., *Parallelized ADI scheme using GECR method and implementation of Navier-Stokes equation in the PAX computer.* August 1985.

10. G. Baudet, "Asynchronous iterative method for multiprocessors," *Journal of the ACM* **25**(2) pp. 226-244 (April 1978).

11. D. Chazan and W. Miranker, "Chaotic relaxation," *Linear algebra and its applications* **2** pp. 199-222 (1969).

12. H. T. Kung, *VLSI, computer science and synergetic research.* February 1983.

13. H. T. Kung and M. S. Lam, *Wafer-scale integration and two-level pipelined implementation of systolic arrays.* January 1984.

14. A. Brandt, "Multilevel adaptive techniques for PDE's: ideas and software," pp. 277-318 in *Mathematical software III,* ed. J. Rice, (1977).

15. A. Brandt, "Multigrid solvers on parallel computers," pp. 39-84 in *Elliptic problem solvers,* ed. M. Schultz, Academic press (1981).

Operator size	Chip count	I/O bandwidth	Throughput
Height x Length		Mword/sec	MOPS
4 x 16	14	1.14/4.58	36.56
8 x 8	20	2.28/2.28	36.56
8 x 32	56	2.28/9.14	145.72
16 x 16	48	4.58/4.58	145.72
8 x 128	200	2.28/36.58	585.16
32 x 32	128	9.14/9.14	585.16

Table 1. Estimated throughput and I/O requirements for different Operator shapes and sizes

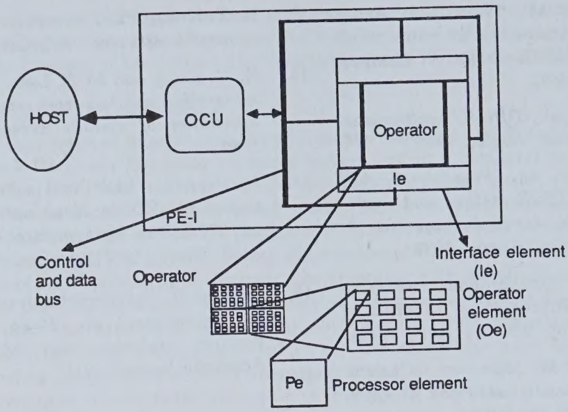

HOST
OCU
Operator
Ie
PE-I
Control and data bus
Operator
Interface element (Ie)
Operator element (Oe)
Pe
Processor element

Fig. 1 Architecture of PE-I

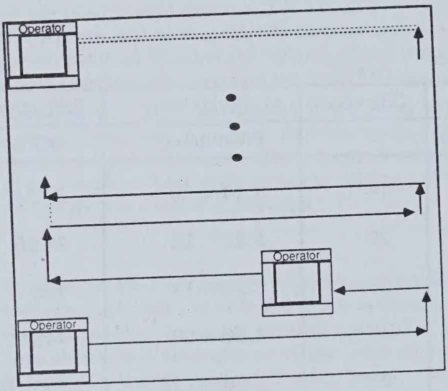

Operator
Operator
Operator

Fig. 2 Movement of the Operator

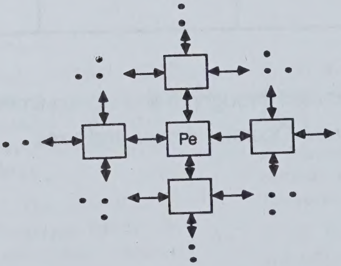


Fig. 3. A sample Pe with its neighbors

CYLINDRICAL ARRAYS FOR MATRIX MULTIPLICATION

W.A. PORTER AND J.L. ARAVENA
Department of Electrical & Computer Engineering
Louisiana State University
Baton Rouge, LA 70803

ABSTRACT

A cylindrical architecture is presented. The computation of matrix products is used to demonstrate the advantage of cylindrical computation verses conventional array computation.

1.0 INTRODUCTION

Recent developments in computer design have witnessed considerable interest in array processing [1], [2], [3], [4]. Array architectures, which can exploit the capabilities of VLSI design, have been touted by many researchers as the best potential solution to future real time large scale system computational requirements.

An important component of the above development is the design of arrays for matrix-matrix and matrix-vector multiplications. Such operations play a key role in many well known algorithms for communications signal extraction and automatic control. One can cite 1-D and 2-D convolutions [5], [6], FFT computations [7], UL factorization [8], matrix triangularization [9] and Riccati equation generation [10] as being indicative of the pervasiveness of matrix operations.

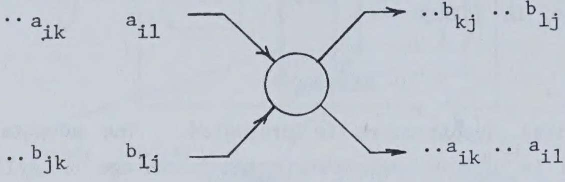
The present study describes an alternative to array processing for matrix products. The cylindrical architecture presented here, is faster than conventional arrays and requires only local broadcasting. There are some tradeoffs, however, which are also summarized.

2.0 DISCUSSION

To place the present development in perspective we summarize briefly a comparable array architecture. For this let A,B,C denote nxn matrices with elements a_{ij}, b_{ij}, c_{ij} respectively. Our objective is to realize the computation C = AB which has the typical element

$$c_{ij} = \sum_{k=1}^{n} a_{ik}b_{kj} , \quad i,j = 1,\dots,n$$

Some clutter can be minimized by assuming that the primitive computational elements, used in the array, can accumulate as well as multiply input signals. Our prototype is depicted in the following figure which depicts also the ordering of the data streams supplied to the element.


·· aik ai1
..bkj .. b1j
..bjk b1j
..aik .. ai1

With no loss of generality we assume that the primitive accumulates the indicated term by term products and transmits upon completion (ie no subtotals are transmitted).

A conventional systolic array is depicted in Figure 2. This array performs matrix multiplication for the 4x4 case. Given the obvious extension this array is also a typical section from the nxn case.

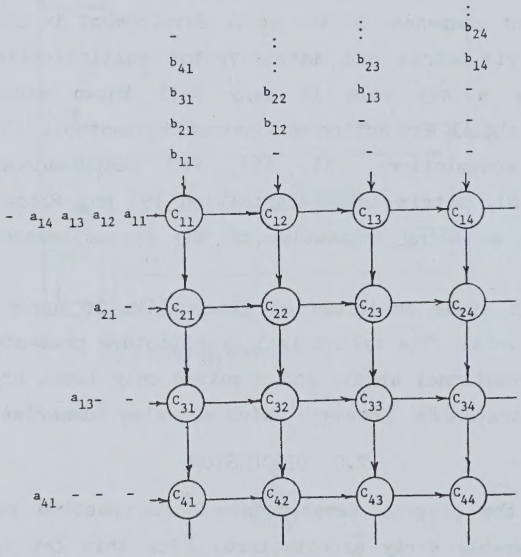

b41
b31
b21
b11
b22
b12
b23
b13
b24
b14
- a14 a13 a12 a11
a21 -
a13- -
a41 - - -
c11
c12
c13
c14
c21
c22
c23
c24
c31
c32
c33
c34
c41
c42
c43
c44

Figure 2

In table 1 we list the input-output activity for the array of Figure 2 (n=4 case) the timing of multiplications and transmissions is not depicted in table. Our step units relate to the completion of the obvious algorithm stages in the computation of the products. One might question, for example, whether c_{11} should be listed at step 4 which coincides with the first availability of a_{14} and b_{41} to the c_{11} node. These issues, however, are non substantial in that we shall continue the same assumptions throughout. As a result comparisons are accurate even though the units are unrefined.

1	a_{11}	b_{11}	
2	a_{12} a_{21}	b_{12} b_{21}	
3	a_{13} a_{22} a_{31}	b_{13} b_{22} b_{31}	
4	a_{14} a_{23} a_{32} a_{41}	b_{14} b_{23} b_{32} b_{41}	c_{11}
5	a_{24} a_{33} a_{42}	b_{24} b_{33} b_{42}	c_{12} c_{21}
6	a_{34} a_{43}	b_{34} b_{43}	c_{13} c_{22} c_{31}
7	a_{44}	b_{44}	c_{14} c_{23} c_{32} c_{41}
8	Input		c_{24} c_{33} c_{42}
9			c_{34} c_{43}
10			c_{44}
			output

Table 1. Array I/O Activity

To describe the cylindrical architecture we envision a cylinder with the primative computing elements placed on its surface in a regular pattern. In Figure 3 we have displayed this cylinder in planar form (n=4 case). The dotted lines indicate proximity on the cylinder that is distorted in the planar model. The computing cylinder could of course be realized in the planar fashion but at the expense of these long connection paths.

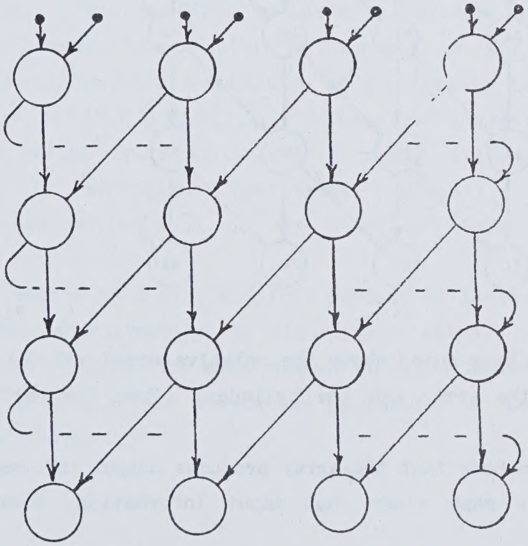


Figure 3

To execute the matrix product C = AB the columns of B are fed at the odd (even) numbered taps while the rows of A are fed to the even (resp. odd) taps one column of A and one row of B at a time. The input sequence mode guarantees that each primitive will perform a valid operation and compute an element of the matrix product. In Fig. 4 we display two input modes and the product element computed by each primitive. In one case the columns of B are fed to the odd taps while in the other they are fed to the even taps. While the cylinder uses the same computational primitives as the array of Figure 2, the transmissions are quite distinct.

With regard to I/O activity we note that the cylinder has a regular input rate which agrees with the maximum input rate of the array. The array computes AB in approximately 3n steps while the cylinder completes the calculation in approximately 2n steps.

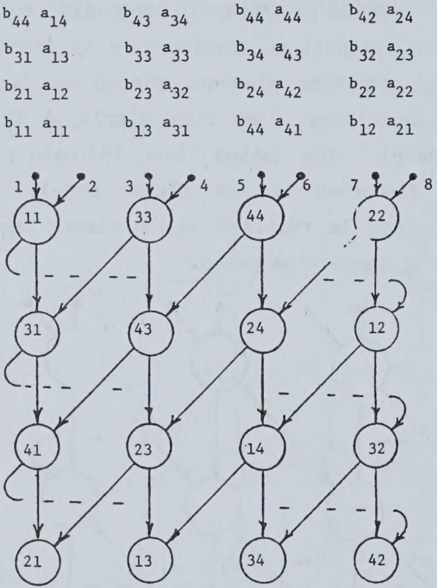

b_{44} a_{14}
b_{43} a_{34}
b_{44} a_{44}
b_{42} a_{24}
b_{31} a_{13}
b_{33} a_{33}
b_{34} a_{43}
b_{32} a_{23}
b_{21} a_{12}
b_{23} a_{32}
b_{24} a_{42}
b_{22} a_{22}
b_{11} a_{11}
b_{13} a_{31}
b_{44} a_{41}
b_{12} a_{21}
1
2
3
4
5
6
7
8
11
33
44
22
31
43
24
12
41
23
14
32
21
13
34
42

Figure 4 a)

3.0 Remarks. We have noted above the relative speed and the relative I/O activity of the array and the cylinder. Some further features deserve mention.

In table 1 we note that the array produces output information, on C in exactly the same order that input information, form A,B is required.

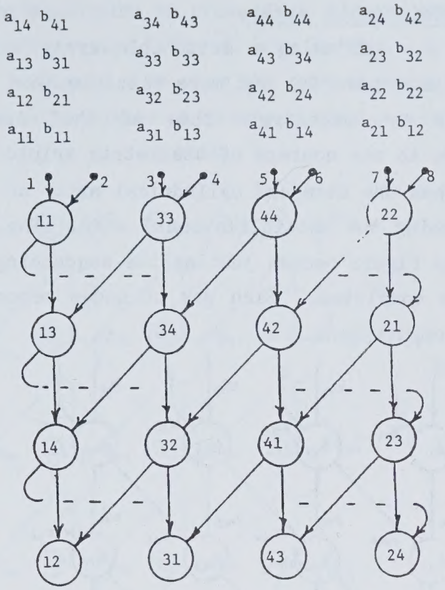

a_{14} b_{41}
a_{13} b_{31}
a_{12} b_{21}
a_{11} b_{11}
a_{34} b_{43}
a_{33} b_{33}
a_{32} b_{23}
a_{31} b_{13}
a_{44} b_{44}
a_{43} b_{34}
a_{42} b_{24}
a_{41} b_{14}
a_{24} b_{42}
a_{23} b_{32}
a_{22} b_{22}
a_{21} b_{12}
1
2
3
4
5
6
7
8
11
33
44
22
13
34
42
21
14
32
41
23
12
31
43
24

Figure 4 b)

This is useful in applications where multiple matrix products are required. Indeed the next product, say KC, can be initiated while C is still under computation.

We note that the cylinder output flow is not identical with the input flow. This possibly requires a reformat stage. Multiple products, however, can still be layered on the cylinder. To illustrate consider the computation of a polynomic function requiring computation of $A^k, k = 0,1,2,\ldots$ The first operation, A^2, must be completed before operation number 2 namily $A(A^2)$can be initiated. However, all subsequent operations, $(A^2)(A^2)$, $A^2(AA^2)\ldots$, can be initiated while each of the previous computations is still underway.

With regard to a single matrix product it is easily seen that an array takes approximately 3n algorithmic steps. The cylindrical architecture computes the same product in approximately 2n algorithmic steps. This single pass speeds are maintained in the interative mode summarized above.

4.0 Using a Switchable Array. In this section we take up the computation of $C = AXB$ using a switchable array. Switchable arrays are in general more powerful and more flexible than fixed arrays. It is of interest to demonstrate some of the capabilities of the switchable array in the context of the matrix triple product.

Consider then the standard cylindrical array of figure (5). This array is performing the multiplication $T = XB$. The data distribution depicted in this figure occurs just as the sequencing of the X,B input data strings is completed. Each row of nodes sequentially completes the indicated computations.

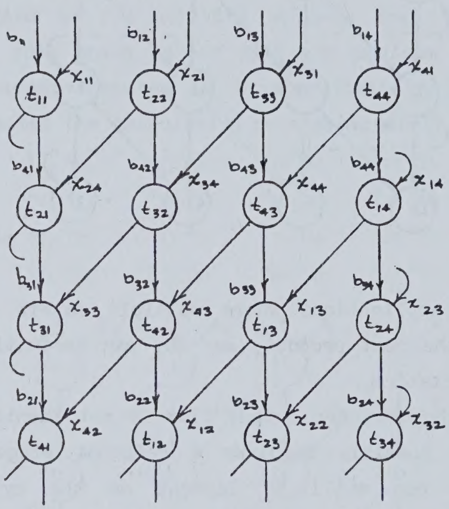

b_{11}
b_{12}
b_{13}
b_{14}
x_{11}
x_{21}
x_{31}
x_{41}
t_{11}
t_{22}
t_{33}
t_{44}
b_{41}
x_{24}
b_{42}
x_{34}
b_{43}
x_{44}
b_{44}
x_{14}
t_{21}
t_{32}
t_{43}
t_{14}
b_{31}
x_{33}
b_{32}
x_{43}
b_{33}
x_{13}
b_{34}
x_{23}
t_{31}
t_{42}
t_{13}
t_{24}
b_{21}
x_{42}
b_{22}
x_{12}
b_{23}
x_{22}
b_{24}
x_{32}
t_{41}
t_{12}
t_{23}
t_{34}

Figure 5

As each row of computations completes the nodes in the row switches to the following form (see figure 6). The value t_{ij} = $(XB)_{ij}$ is fixed in a register at the node in question. The spiral data stream is subsequently multiplied by the register value. The node no longer accumulates but rather added the $a_{ij}t_{jk}$ value to the incoming value on the downward path and transmits this partial sum downward.

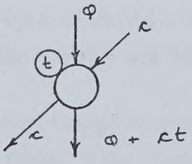

φ
r
t
r
$\varphi + rt$

Figure 6

As for the array inputs we proceed as follows the B data strings are followed by zero strings. The X data strings are followed by the A data strings. In Figure 7 we show the data distribution as the data strings terminate. The number of primes on the partial sums indicates the number of terms yet to be added.

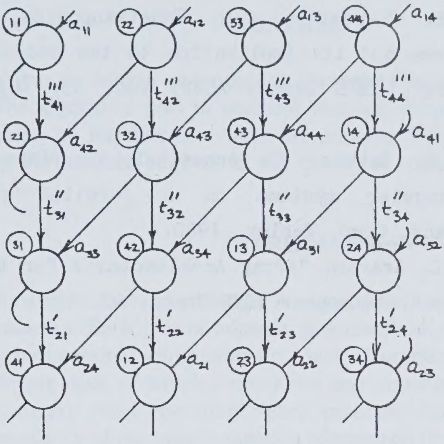

11
a_{11}
22
a_{12}
33
a_{13}
44
a_{14}
t'''_{41}
t'''_{42}
t'''_{43}
t'''_{44}
21
a_{42}
32
a_{43}
43
a_{44}
14
a_{41}
t''_{31}
t''_{32}
t''_{33}
t''_{34}
31
a_{33}
42
a_{34}
13
a_{31}
24
a_{32}
t'_{21}
t'_{22}
t'_{23}
t'_{24}
41
a_{24}
12
a_{21}
23
a_{22}
34
a_{23}

Figure 7

The switchable array outputs the triple product at a uniform rate. Moreover this output occurs at fixed locations on the array. The triple product computation is completed in 3n steps.

REFERENCES

[1] Guo-Jie, L. and B. Wah, "The Design of Optimal Systolic Arrays", IEEE Trans. on Comp., Vol. 34, No. 1, Jan. 1985, pp. 66-77.

[2] P.R. Cappello and K. Steiglitz, "Unifying VLSI array designs with geometric transformations," in Proc. Int. Conf. Parallel Processing, 1983, pp. 448-457.

[3] J.A.B. Fortes, "Algorithm transformations for parallel processing and VLSI architecture design," Ph.D. dissertation, Univ. Southern California, Los Angeles, CA, Dec. 1983.

[4] K. Hwang and Y.H. Cheng, "VLSI computing structure for solving large scale linear system of equations," in Proc., Int. Conf. Parallel Processing, Aug. 1980, pp. 217-227.

[5] Kung, H.T., L.M. Ruane D.W.L. Yen, "A Two level pipelined systolic array for convolutions:. VLSI Systems and Computations, pp. 255-264, Computer Science Dept. Carnegie-Mellon, 1981.

[6] Liu, K.Y., "A pipelined tree machine architecture for computing a multidimensional convolution". IEEE Trans. CAS, vol. 29, NO. 4, April 1982.

[7] Wold, E.H., A.M. Despain, "Pipeline and parallel-pipeline FFT processors for VLSI implementation". IEEE Trans. Comp., Vol. C-33, No. 5, May 1984.

[8] Nikias, C.L., A.P. Chrysafis, A.N. Venetsanpoulos, "The LU decomposition theorem and its implication to the realization of 2-D digital filters", IEEE Trans. ASSP, Vol. 33, No. 3, June 1985.

[9] Montoye, R.K., D.H. Lawrie, "A practical algorithm for the solution of triangular systems on a parallel processing systems". IEEE Trans. Comp.,Wesley, 1980.

[10] W.A. Porter and J.L. Aravena "Array Architectures for Estimation and Control Function", to appear IEEE Trans. AC. 1986.

SORTING IN CONSTANT NUMBER OF ROW AND COLUMN PHASES ON A MESH†

JOHN M. MARBERG and ELI GAFNI
Computer Science Department
University of California
Los Angeles, CA 90024

ABSTRACT

An algorithm for sorting on a mesh by alternately transforming the rows and columns is presented. The algorithm runs in constant number of row- and column-transformation phases (16 phases), an improvement over the previous best upper bound of $O(\log\log m)$ phases, m being the number of rows in the mesh. A corresponding lower bound of five phases is also shown.

1. INTRODUCTION

A two-dimensional *mesh* consists of $m \cdot n$ processors organized in a rectangular grid of m rows and n columns. Each pair of adjacent processors along the same row or column are connected by a bidirectional communication link. Computation proceeds in synchronous parallel steps, during each of which a processor may transfer a value to a neighbor or perform some elementary local operation. Each processor has a constant amount of local memory. The regularity of the interconnection pattern and the modularity of the layout make the mesh an ideal model for VLSI applications. It is therefore that considerable attention has been given to the study of algorithms for this model.

In this paper we consider the problem of sorting on a mesh. It is defined as follows: a collection of elements is distributed in the mesh, one element per processor; viewing the input as an $m \times n$ matrix, sort the matrix into row-major order. Two commonly used row-major orders are: (row, column)-lexicographic, and snake-like [Thom77]. We develop a sorting algorithm that works by alternately transforming the rows and columns of the mesh. This approach is particularly attractive for VLSI, since it lends itself easily to systolic implementation.

Recent results by Bilardi and Preparata [Bila85] and by Leighton [Leig85] have established the VLSI optimality of the AKS sorting network [Ajta83]. Yet, the enormous constants involved in the AKS network render it largely impractical. This further motivates the effort to find simple sorting methods for regular interconnection structures such as the mesh.

The asymptotic lower bound for sorting on a mesh is $\Omega(n+m)$ parallel steps, established by Thompson and Kung [Thom77]. Numerous researchers have developed optimal algorithms, including Thompson and Kung [Thom77], Nassimi and Sahni [Nass79], Kumar and Hirschberg [Kuma83], Lang, Schimmler, Schmeck and Schroder [Lang85], and Schnorr and Shamir [Schn86]. All the above-mentioned algorithms proceed by dividing the mesh into submeshes, working on the submeshes recursively in parallel, then combining the results in some fashion. Recently, Scherson, Sen and Shamir [Sche86] have discovered an algorithm that is based on an entirely different approach. It proceeds by alternately sorting the rows and columns of the mesh, as follows.

---

† Work supported by an NSF PYI Award and by an IBM Faculty Development Award.

Algorithm *SHEARSORT* [Sche86]

Repeat $\lceil \log m \rceil + 1$ times steps 1 and 2 below.

1. Sort all even-numbered rows to the right and all odd-numbered rows to the left.
2. Sort all the columns downwards.

This novel algorithm is considerably simpler than previous solutions to the sorting problem. The only computation required is sorting along a line, which can be done using the straightforward odd/even transposition method [Knut73]. Consequently, the algorithm entails relatively little control logic for the routing of data. Moreover, the row/column approach lends itself naturally to systolic implementation. Notice, however, that *SHEARSORT* is not optimal; consisting of $\log m$ sorting phases, it requires a total of $O((m+n)\log m)$ steps. Schnorr and Shamir [Schn86] have subsequently developed an improved version of the algorithm, called *REVSORT*, which runs in $\log\log m$ sorting phases and a total of $O((m+n)\log\log m)$ steps.

Let us define a *row-phase* of a mesh algorithm as a maximal sequence of steps in which all the data movement is along the rows. A *column-phase* is defined similarly. Thus, a row-phase (column-phase) results in some arbitrary transformation of each row (column). An algorithm that operates in alternating row- and column-phases is called a *row/column* algorithm. An open question posed in [Schn86] is whether there exists a row/column sorting algorithm with fewer than $O(\log\log m)$ phases (which is the number of phases in *REVSORT*). In this paper we answer this question affirmatively. Specifically, we develop an algorithm that runs in a constant number of phases (16 phases), requiring a total of $O(m+n)$ steps, which is optimal. We use only two types of transformations during a phase: sorting and rotation. Since rotation can be emulated by sorting, the algorithm can be implemented entirely by means of basic compare/exchange steps. We also prove that five phases is a lower bound for row/column sorting. Our results demonstrate the viability of the row/column approach, providing motivation to investigate its applicability to other problems.

2. THE SORTING ALGORITHM

We denote the positions in the mesh by the (row, column) pairs (i, j), where $0 \le i \le m-1$ and $0 \le j \le n-1$. For simplicity, we assume $m=2^s$ and $n=2^{2t}$, where $s \ge t$ (the results can be generalized for other values of m and n).

A *slice* is a submesh of size $n^{1/2} \times n$, consisting of all rows i such that $kn^{1/2} \le i \le (k+1)n^{1/2}-1$ for some given $k \ge 0$. A *block* is a submesh of size $n^{1/2} \times n^{1/2}$, consisting of all positions (i, j) such that $kn^{1/2} \le i \le (k+1)n^{1/2}-1$ and $ln^{1/2} \le j \le (l+1)n^{1/2}-1$ for some given $k \ge 0$ and $l \ge 0$. Figure 1 illustrates these definitions.

It is convenient to analyze the sorting algorithm via the 0-1 principle [Knut73]. This principle states that if a sorting algorithm works correctly on an arbitrary input of 0's and 1's, then it works correctly on any input. Given a mesh with 0-1 input, we say that a given region (e.g., row, slice, block) is *dirty* if it contains both 0's and 1's; otherwise it is *clean*. The correctness of a sorting algorithm can be shown by identifying the regions of the mesh that become dirty or clean at any given point. Clearly, upon termination, there should remain at most one dirty row.

We begin by specifying several macros (or procedures), which serve as building-blocks for the main algorithm. Each macro comprises a sequence of phases that accomplish a specific transformation on the mesh.

Macro *BALANCE*

Applied to a (sub)mesh of size $v \times w$, this macro "balances" the distribution of 0's and 1's among the columns, thereby reducing the number of dirty rows to at most $\min\{v, w\}$. This is accomplished by "dealing" the sorted elements of each column in round-robin fashion to all the columns. The macro consists of the following three steps.

(a) Sort all the columns downwards.

(b) Rotate each row i $(i \bmod w)$ positions to the right.

(c) Sort all the columns downwards.

Macro *UNBLOCK*

This macro distributes the elements of each block among all the columns (i.e., it "unblocks" the blocks), thereby effectively generating a clean row from every clean block, and a (possibly) dirty row from every dirty block. A similar idea is used in the *REVSORT* algorithm [Schn86], except that the goal there is to "un-rectangle" arbitrary rectangles of area n, not just square blocks. The macro consists of the following two steps.

(a) Rotate each row i $(i \bmod n^{1/2})$ positions to the right.

(b) Sort all the columns downwards.

Macro *SHEAR*

This macro emulates one iteration of the *SHEARSORT* algorithm [Sche86]. Assuming that all dirty rows are in one contiguous region, each pair of adjacent dirty rows are reorganized into one clean row and one (possibly) dirty row, thereby cutting the number of dirty rows by half. This is effectively accomplished by sorting alternate rows in opposite directions (i.e., "shearing" the rows). The macro consists of the following two steps.

(a) Sort all even-numbered rows to the right and all odd-numbered rows to the left.

(b) Sort all the columns downwards.

We now specify the main algorithm, called *ROTATESORT*, using the above macros. A complete proof of correctness and complexity analysis are given in Section 3.

Algorithm *ROTATESORT*

1. *BALANCE* the mesh.
   ⟨ now there are at most n dirty rows ⟩
2. Sort all the rows to the right.
   ⟨ now there are at most $2n^{1/2}$ dirty blocks ⟩
3. *UNBLOCK* the mesh.
   ⟨ now there are at most $2n^{1/2}$ dirty rows ⟩
4. *BALANCE* each slice as if it were an $n \times n^{1/2}$ mesh lying on its side.
   ⟨ now there are at most 6 dirty blocks ⟩
5. *UNBLOCK* the mesh.
   ⟨ now there are at most 6 dirty rows ⟩
6. Repeat macro *SHEAR* three times.
   ⟨ now there is at most one dirty row ⟩
7. Sort all the rows to the right to obtain lexicographic order, or alternating rows in opposite directions to obtain snake-like order.

3. CORRECTNESS AND COMPLEXITY

To prove the correctness of *ROTATESORT* using the 0-1 principle, we need to show that upon termination:

(a) there exists at most one dirty row;

(b) the dirty row is sorted in accordance with the choice of row-major order;

(c) the dirty row separates the clean rows of 0's from the clean rows of 1's.

We first prove the correctness of the three macros. Once the macros are validated, it is straightforward to prove the main algorithm.

Lemma 1. After applying macro *BALANCE* to a mesh of size $v \times w$, there remain at most $\min\{v, w\}$ dirty rows.

Proof. The claim is trivial when $v \leq w$. We now discuss the case $v > w$. Step (b) of *BALANCE* has the effect of "dealing" the elements of each column in round-robin fashion among all the columns. Since the columns were previously sorted in step (a), the difference between the number of 0's that any two columns receive from a given column is at most one. Thus, the total difference in the number of 0's between any two columns after step (b) is at most w. Sorting the columns in step (c) thus yields at most w dirty rows. Moreover, all the resulting dirty rows are in one contiguous region that separates the clean rows of 0's from the clean rows of 1's (see Figure 2). ■

Lemma 2. After applying macro *UNBLOCK* to a mesh with k dirty blocks, there remain at most k dirty rows.

Proof. Step (a) of *UNBLOCK* moves each of the n elements of a block into a different column. Since there are k dirty blocks, the difference in the number of 0's between any two columns after step (a) is at most k. The Lemma follows since the columns are subsequently sorted in step (b). As in Lemma 1, the dirty rows are in one contiguous region that separates the clean rows of 0's from the clean rows of 1's. ■

Lemma 3. After applying macro *SHEAR* to a mesh with k dirty rows, there remain at most $\lceil \frac{k}{2} \rceil$ dirty rows.

Proof. The proof is based on [Sche86]. We assume that upon invocation of *SHEAR* all the dirty rows are in one contiguous region. Consider a pair of adjacent dirty rows. After step (a), the 0's are packed in opposite ends in the two rows (see Figure 3). Let z_1 and z_2 denote the number of 0's in the rows. If $z_1 + z_2 \geq n$, then there is at least one 0 in each column in the pair; otherwise, there is at least one 1 in each column. In either case, the difference between the number of 0's in any two columns in the pair is at most one. Since there are $\lceil \frac{k}{2} \rceil$ disjoint pairs of adjacent dirty rows (when k is odd, one pair has only one dirty row), the total difference in the number of 0's between any two columns in the mesh at the end of step (a) is at most $\lceil \frac{k}{2} \rceil$. The Lemma follows since the columns are subsequently sorted in step (b). As in Lemmas 1 and 2, the resulting dirty rows are in one contiguous region. ■

Theorem 1. Statements 1-7 below describe the configuration of the mesh after steps 1-7, respectively, of algorithm *ROTATESORT*.

1. After step 1 there are at most n dirty rows.
2. After step 2 there are at most $2n^{1/2}$ dirty blocks.

3. After step 3 there are at most $2n^{1/2}$ dirty rows.
4. After step 4 there are at most 6 dirty blocks.
5. After step 5 there are at most 6 dirty rows.
6. After step 6 there is at most one dirty row.
7. After step 7 the mesh is sorted.

Proof.

1. The result follows immediately from Lemma 1.
2. The proof is based on [Schn86]. Notice that at the end of step 1 the columns are sorted. Sorting the rows in step 2 thus results in the following configuration (see Figure 4): all the 0's are in the upper-left corner of the mesh, whereas the 1's are in the lower-right. The number of dirty rows remains unchanged, i.e., at most n. However, the boundary between the 0's and 1's is now monotonic (in the sense that it does not cross the same row or column twice), and therefore traverses at most $2n^{1/2}$ blocks. Clearly, the blocks traversed by the boundary are dirty, whereas the remaining blocks are clean.
3. The result follows immediately from claim 2 of this Theorem and from Lemma 2.
4. The $2n^{1/2}$ dirty rows that remain at the beginning of step 4 are in one contiguous region (Lemma 2). Hence, there are at most three dirty slices in the mesh. We now view each slice as a mesh of size $n \times n^{1/2}$ lying on its side, and apply *BALANCE* to it. By Lemma 1, this reduces the number of dirty "rows" in each inverted slice to at most $n^{1/2}$. Moreover, the dirty "rows" in each slice are in one contiguous region (see Figure 5). Hence, at the end of step 4 there are at most two dirty blocks in each dirty slice, for a total of six dirty blocks in the entire mesh.
5. The result follows immediately from claim 4 of this Theorem and from Lemma 2.
6. By Lemma 3, each application of macro *SHEAR* reduces the number of dirty rows in the mesh by half. Thus, after $\lceil \log_2 6 \rceil = 3$ applications, there remains at most one dirty row. Moreover, the dirty row separates the clean rows of 0's from the clean rows of 1's. Hence, termination requirements (a) and (b) are satisfied after step 6.
7. Step 7 sorts each row in accordance with the choice of row-major order, thereby satisfying termination requirement (c). This completes the correctness proof. ■

We now discuss the complexity of *ROTATESORT*. It is easy to see that the algorithm runs in constant number of phases. Each application of *BALANCE* takes three phases, whereas *UNBLOCK* and *SHEAR* each take two phases. Summing over the entire algorithm, one would get a total of 18 phases. Yet, a closer look at the sequence of steps reveals that steps 2 and 3(a) both operate on rows, and hence, by definition, constitute a single phase. The same observation holds for steps 4(c) and 5(a). The correct number of phases is therefore 16.

The algorithm uses only two types of transformations during a phase: sorting and rotation. Since rotation can be emulated by sorting, each row-phase can be implemented in $O(n)$ basic compare/exchange steps, using odd/even transposition sort [Knut73]. Similarly, a column-phase can be implemented in $O(m)$ compare/exchange steps. Thus, the total number of basic steps in the entire algorithm is $O(m+n)$, which is optimal.

It can be seen from Lemma 1 that when $m \leq n$, the application of *BALANCE* in step 1 is redundant and can be replaced with a single phase that sorts all the columns downwards. Consequently, for $m \leq n$, and in particular for square meshes, we obtain an algorithm with 14 phases and $O(n)$ basic steps.

4. A LOWER BOUND

An obvious question to be posed at this point is whether *ROTATESORT* is optimal in the number of phases. In an effort to answer this question, we show that five phases is a lower bound. However, we suspect that neither this lower bound, nor the upper bound of 16 phases are tight.

The lower bound argument proceeds by constructing for any given algorithm an input that requires five phases. We assume that the algorithm is comparison-based, i.e., that the values of the elements are used only in comparisons with each other. The bound is valid regardless of the sorting order being used (i.e., not just for row-major order).

We use the following notation. Two positions (i_1, j_1) and (i_2, j_2) in the mesh are *relatively independent* if $i_1 \neq i_2$ and $j_1 \neq j_2$. Position (i, j) is said to have *row-rank* r at a given point in the algorithm, if at that point the element located in (i, j) is the r-th smallest element in row i. The term *column-rank* is defined similarly. The row-rank and column-rank of position (i, j) at the end of phase t, $t \geq 1$, are denoted $RR_t(i, j)$ and $CR_t(i, j)$, respectively.

We assume without loss of generality that all elements are distinct. The i-th smallest element, $1 \leq i \leq mn$, is denoted e_i.

The property of row/column sorting upon which the proof of the lower bound hinges is the following.

Lemma 4. Let e_a and e_b be two elements that belong in the same column (in the same row, respectively) in the sorted order. If at the end of a given column-phase t (row-phase t) both e_a and e_b are located in the same row (same column), then the algorithm requires at least t+3 phases.

Proof. We only prove the case where e_a and e_b belong in the same column in the sorted order. The other case can be shown by reversing the roles of row and column.

In order to move e_a and e_b from the same row to the same column after phase t, we need at least one more column-phase followed by a row-phase. Since phase t+1 is a row-phase, at least 3 additional phases are required. ■

To obtain the lower bound of five phases, we construct an input such that the premise of Lemma 4 is satisfied at the end of phase 2 (clearly, two phases are always necessary).

Theorem 2. Any row/column sorting algorithm on a mesh of size $m \times n$, where $m \geq 8$, $n \geq 8$, requires at least five phases.

Proof. We distinguish between an algorithm that begins with a row-phase, and one that begins with a column-phase. We discuss only the former case. By symmetry, the other case can be shown by reversing the roles of row and column.

Given that the first phase is a row-phase, we show how to construct, for any given sorting algorithm, an input such that two elements which belong in the same column in the sorted order are located in the same row at the end of the phase 2.

Let l be some fixed integer, $l \leq mn - \max\{m, n\}$, called the *partition index*. We partition the input into three different classes with respect to l: elements e_i such that $i < l$ are called *very small*; elements e_i such that $l \leq i \leq l + \max\{m, n\}$ are called *intermediate*; the remaining elements, i.e., elements e_i such that $i > l + \max\{m, n\}$ are called *very large*. Regardless of the actual value of l, one can always identify three intermediate elements: e_{p_1}, e_{p_2} and e_{p_3}, such that e_{p_1} and e_{p_2} belong in the same column in the sorted order, whereas e_{p_3} belongs in a different column. This is because the number of intermediate elements ($\max\{m, n\}+1$) is larger than the number of rows or columns. We call each of e_{p_1}, e_{p_2} and e_{p_3} a *pivot*.

Consider an input I such that at the end of phase 1 the position of each pivot is independent from the position of any other intermediate element (including the other two pivots). In other words, no pivot "sees" any of the other intermediate elements during phases 1 and 2. Now, let us form another input by permuting the pivots in I among themselves in some arbitrary fashion, leaving all the remaining positions unchanged. It can be seen that at the end of phase 2 each element other than the pivots will occupy the exact same position under any such permutation, whereas the pivots will be permuted. This is because the three pivots have the same relative order with respect to any very small or very large element, thus rendering a permutation of the pivots transparent to the algorithm during the first two phases.

We now prove that the premise of Lemma 4 is satisfied at the end of phase 2 by some of the $3!=6$ possible pivot permutations of input I. Let S denote the set of elements comprising the three pivots plus the $m-2$ non-pivot elements that in the sorted order belong in the same column as e_{p_1} and e_{p_2}. The positions occupied by the elements in S at the end of phase 2 are fixed up to permutation among the pivot positions. Moreover, since there are $m+1$ elements in S but only m rows, two of the elements in S must be in the same row at all times, particularly at the end of phase 2. In other words, there exist two fixed positions, both in the same row, that contain elements from S at the end of phase 2 under all six permutations. Now, e_{p_3} can occupy one of the two positions in at most four permutations. This leaves two permutations in which both positions are occupied by elements from $S-\{e_{p_3}\}$. Since these elements belong in the same column in the sorted order, the premise of Lemma 4 is satisfied.

To complete the proof of the theorem, we need to show how to construct input I. Let us assume, for now, that the partition index l is a parameter. Later, we will show how to determine its exact value.

First, we arbitrarily fix the order among the elements in each row of the input. This enables us to uniquely determine, for a given algorithm, the row-rank of every position at the end of phase 1. Notice that we are still left with complete freedom to decide which element goes in which row. As will be seen below, knowing all the RR_1 values suffices in order to construct the proper input.

We construct the input row by row, first taking care of the rows that contain pivots. To this end, we need to guarantee that at the end of phase 1 the pivots will be in independent positions. Let (r_1, c_1), (r_2, c_2), and (r_3, c_3) be three pairwise independent positions. To each row r_j, $1 \le j \le 3$, we allocate the following elements:

(a) e_{p_j};

(b) $RR_1(r_j, c_j)-1$ very small elements;

(c) $n-RR_1(r_j, c_j)$ very large elements.

It can be seen that e_{p_j} is the element of rank $RR_1(r_j, c_j)$ in its row. Hence, it will be in position (r_j, c_j) at the end of phase 1.

We now construct the rows that contain the remaining intermediate elements. We must make sure that no intermediate element will be in column c_1, c_2 or c_3 at the end of phase 1. Let us consider an arbitrary empty row k. We assume without loss of generality that $RR_1(k, c_1) \le RR_1(k, c_2) \le RR_1(k, c_3)$ (if not, simply "rename" the columns). Since the number of elements in a row is n, at least one of the following four inequalities must always hold.

(a) $RR_1(k, c_1)-1 \geq \frac{n-3}{4}$

(b) $RR_1(k, c_2)-RR_1(k, c_1)-1 \geq \frac{n-3}{4}$

(c) $RR_1(k, c_3)-RR_1(k, c_2)-1 \geq \frac{n-3}{4}$

(d) $n-RR_1(k, c_3) \geq \frac{n-3}{4}$

Thus, it is always possible to allocate $\frac{n-3}{4}$ intermediate elements to row k. For example, assume inequality (b) holds. First, we allocate to row k $RR_1(k, c_1)$ very small elements and $n-RR_1(k, c_2)+1$ very large elements. This leaves room for at least $\frac{n-3}{4}$ additional elements. Any intermediate element we now allocate to row k must, by definition, have a rank (among the elements in the row) which is larger than $RR_1(k, c_1)$ and smaller than $RR_1(k, c_2)$. This guarantees that at the end of phase 1 such an element will not be in any of the columns containing a pivot. Inequalities (a), (c) and (d) are handled in a similar way. Thus, all the $\max\{m, n\}-2$ intermediate elements that are not pivots can be accommodated in $\lceil \frac{4(\max\{m, n\}-2)}{n-3} \rceil$ rows.

This takes care of all the rows that contain intermediate elements. The remaining rows, if any, can be constructed arbitrarily from the remaining elements.

The construction imposes the following constraints on the size of the mesh. We need three rows for the pivots, and as many as $\lceil \frac{4(\max\{m, n\}-2)}{n-3} \rceil$ rows for the remaining intermediate elements. Thus, the inequality $m \geq \frac{4(\max\{m, n\}-2)}{n-3}+3$ must hold.[1] It can be seen that this occurs when $m \geq 8$, $n \geq 8$.

It remains to fix the value of l, the partition index. Let u and v be the total number of very small and very large elements, respectively, allocated to rows containing intermediate elements. l can be set to any integer in the range $[u+1, mn-\max\{m, n\}-v]$. ■

5. CONCLUSION

We have presented an optimal algorithm for sorting on a mesh by means of alternating row and column transformations. Our results demonstrate that the row/column approach is viable, providing motivation to investigate its applicability to other problems.

An interesting open question is to determine the VLSI complexity of *ROTATESORT*. Another issue that remains open is to bridge the gap between the upper bound of 16 phases and the lower bound of five phases. We suspect that neither bound is tight. Specifically, the large degree of freedom in the construction in Theorem 2 suggests that there is room for improvement of the lower bound. Also, it might be possible to reduce the upper bound by using more powerful transformations.

---

[1] The $\lceil \, \rceil$ operator is not needed in the inequality since m is always an integer.

REFERENCES

[Ajta83] Ajtai, M., J. Komolos, and E. Szemeredi, ''An $O(N \log N)$ Sorting Network,'' in *Proceedings 15th ACM Symp. on Theory of Computing*, 1983, pp. 1-9.

[Bila85] Bilardi, G. and F.P. Preparata, ''The VLSI Optimality of the AKS Sorting Network,'' *Information Processing Letters* **20**, 2 (Feb. 1985), pp. 55-59.

[Knut73] Knuth, D.E., *The Art of Computer Programming, Vol. 3: Sorting and Searching*, Addison Wesley, Reading, MA, 1973.

[Kuma83] Kumar, M. and D.S. Hirschberg, ''An Efficient Implementation of Batcher's Odd-Even Merge Algorithm and Its Application in Parallel Sorting Schemes,'' *IEEE Trans. on Computers* **C-32**, 3 (March 1983), pp. 254-264.

[Lang85] Lang, H.W., M. Schimmler, H. Schmeck, and H. Schroder, ''Systolic Sorting on a Mesh-Connected Network,'' *IEEE Trans. on Computers* **C-34**, 7 (July 1985), pp. 652-658.

[Leig85] Leighton, T., ''Tight Bounds on the Complexity of Parallel Sorting,'' *IEEE Trans. on Computers* **C-34**, 4 (April 1985), pp. 344-354.

[Nass79] Nassimi, D. and S. Sahni, ''Bitonic Sort on a Mesh-Connected Parallel Computer,'' *IEEE Trans. on Computers* **C-28**, 1 (Jan. 1979), pp. 2-7.

[Sche86] Scherson, I.D., S. Sen, and A. Shamir, ''Shear Sort: A True Two-Dimensional Sorting Technique for VLSI Networks,'' in *Proceedings 1986 Int. Conf. on Parallel Processing*, pp. 903-908.

[Schn86] Schnorr, C.P. and A. Shamir, ''An Optimal Sorting Algorithm for Mesh Connected Computers,'' in *Proceedings 18th ACM Symp. on Theory of Computing*, 1986, pp. 255-261.

[Thom77] Thompson, C.D. and H.T. Kung, ''Sorting on a Mesh-Connected Parallel Computer,'' *CACM* **20**, 4 (April 1977), pp. 263-271.

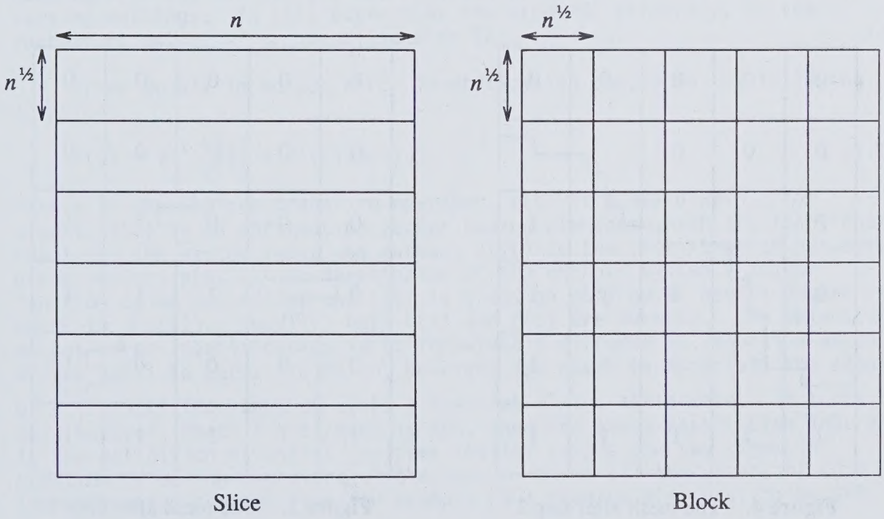


Figure 1. Definition of slice and block

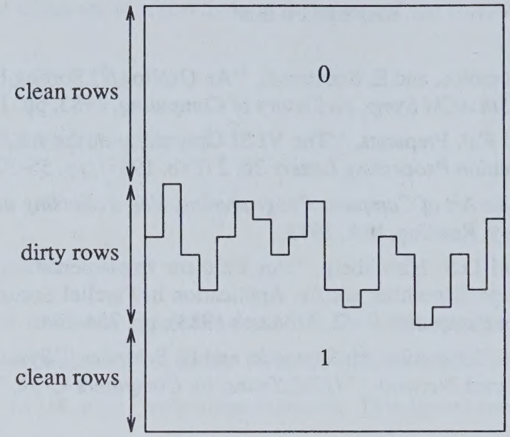

0
clean rows
dirty rows
1
clean rows

Figure 2. Clean and dirty regions after sorting the columns

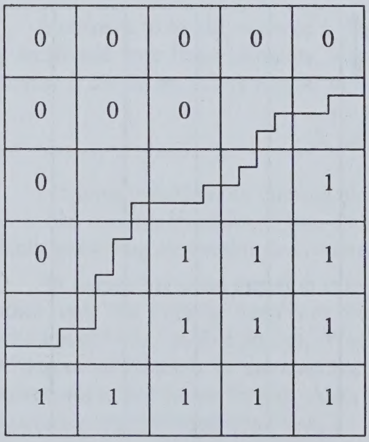
0	0	0	0	0	0	0	1	1	1
1	1	1	1	1	0	0	0	0	0

$z_1+z_2 \geq n$

0	0	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	0

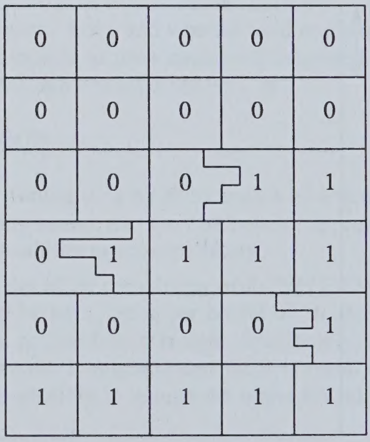
$z_1+z_2<n$

Figure 3. Two adjacent dirty rows sorted in opposite directions



0	0	0	0	0
0	0	0		
0				1
0		1	1	1
		1	1	1
1	1	1	1	1

Figure 4. The mesh after step 2



0	0	0	0	0
0	0	0	0	0
0	0	0	1	1
0	1	1	1	1
0	0	0	0	1
1	1	1	1	1

Figure 5. The mesh after step 4

CONDITIONS FOR DESIGNING STRICTLY POSITIVE REAL TRANSFER FUNCTIONS FOR ADAPTIVE OUTPUT ERROR IDENTIFICATION

SOURA DASGUPTA and AJAY S. BHAGWAT
Department of Electrical and Computer Engineering
University of Iowa, Iowa City, Iowa 52242 USA

ABSTRACT

Strictly Positive Real (SPR) transfer funtions are of importance in Adaptive System Theory. This paper addresses two issues, both related to the problem of finding a polynomial b(s) such that b(s)/p(s) is SPR for all p(s) belonging to a set of nth degree Hurwitz polynomials defined by the magnitude bounds on the coefficients of p(s). It is shown that irrespective of n there exist four polynomials belonging to this set which have the property that if the ratios of b(s) with each of these is SPR then b(s)/p(s) is SPR for all p(s) belonging to the above mentioned set. Moreover, a sufficient condition is derived on the extent of this set, for which a b(s) satisfying this property exists, and a method of constructing b(s) is outlined. The condition in question resembles the Routh array condition for determining system stability.

1. INTRODUCTION

This paper investigates issues related to the design of strictly positive real (SPR) transfer functions useful in adaptive systems theory. For the purposes of this paper a transfer function $G(s) = p(s)/q(s)$ is defined to be strictly positive real if both p(s) and q(s) are Hurwitz polynomials and $\mathrm{Re}[G(j\omega] > 0\ \forall\omega\varepsilon R$. It is important to note that if G(s) is SPR then so is $G^{-1}(s)$.

The importance of SPR transfer functions was first identified by Parks [1]. Thereafter several others including Landau [2], Narendra and coworkers [3-5], Johnson [6] and Anderson [7-8] have used them under varying settings. In this paper they are studied, primarily, in the context of output error identification [2].

Error models in output error identification reduce to the following equation

$$e(t) = \pounds^{-1}\left[\frac{b(s)}{p(s)}\ \pounds(V^T(t)\Delta\Theta(t))\right] \tag{1.1}$$

Here £ is the Laplace transform operator, e(t) is a measurable error signal, V(t) is an information vector having elements which are functionals involving the system input and output, $\Delta\Theta(t)$ is the instantaneous parameter error vector, p(s) is the denominator of the unknown system transfer function to be identified and b(s) is a design polynomial having degree equal to $\delta[p(s)]$. Usually, both b(s) and p(s) are Hurwitz. The object in output error identification is to formulate a differential equation which drives $\Delta\Theta(t)$ to zero. To do so, however, one needs to ascertain the sign of $V^T(t)\Delta\Theta(t)$ from that of e(t); a task not always achievable. It turns out, however, that if b(s)/p(s) is SPR, then the phase restriction inherent in the definition of strict positive reality causes the two signs in question to be "on the average" the same. Indeed all the proofs of such identification schemes (see for example [9]) require b(s)/p(s) to be SPR.

Thus we have to design a polynomial $b(s)$ with $\delta[b(s)] = \delta[p(s)]$ such that $b(s)/p(s)$ is SPR. But the principle difficulty in the design is that $p(s)$ itself is unknown, being the denominator polynomial of the system to be identified.

Infact apart from the knowledge of its degree the only assumptions that can reasonably be made about $p(s)$ are:

(i) that bounds on its coefficients are known

and

(ii) that all polynomials belonging to the set defined by these bounds are Hurwitz.

Define this set to be $\P^n$. Formally, $\P^n$ is the set of real, Hurwitz polynomials $p(s)$ such that

$$p(s) = s^n + p_1 s^{n-1} + p_2 s^{n-2} + \cdots + p_n \tag{1.2}$$

where $0 < \alpha_i < p_i < \beta_i \quad \forall i=1,2,\cdots,n$

Accordingly the issues addressed are two-fold. First, given the set $\P^n$ and a candidate polynomial $b(s)$, how to check whether $b(s)/p(s)$ is SPR $\forall p(s) \in \P^n$? we show that there are only four polynomials in $\P^n$ such that if the ratios of $b(s)$ with each of these are SPR then $b(s)/p(s)$ is SPR $\forall p(s) \in \P^n$.

Second, given the set $\P^n$ what constitute sufficient conditions for the existence of a $b(s)$ such that $b(s)/p(s)$ is SPR $\forall p(s) \in \P^n$? To this end we develop a test similar to the Routh array test on the coefficient bounds defining this set. We construct two arrays similar in structure and construction to the well known Routh array. It is shown that a sufficient condition for the existence of $b(s)$ is that all the elements appearing in the first columns of these arrays be positive. The proof also gives a procedure for constructing $b(s)$.

In the next two sections we address the two issues mentioned above. Section 4 makes the concluding remarks. Most of the proofs have been omitted due to space constraints, and can be found in [16].

2. CHECKING FOR STRICT POSITIVE REALITY

In this section we address the problem of checking $b(s)/p(s)$ for strict positive reality $\forall p(s) \in \P^n$ and a given candidate polynomial $b(s)$. Potentially there are an infinite number of transfer functions which need to be checked. However, we show that irrespective of n, there exist four polynomials $K_i(s)$, $i=1,2,3,4$ in the set $\P^n$ such that if $K_i(s)/b(s)$ is SPR $\forall i=1,2,3,4$ then $p(s)/b(s)$ is SPR $\forall p(s) \in \P^n$, whence $b(s)/p(s)$ is also SPR.

Define $\P_c^n$ to be the set of 2^n polynomials given by (1.2) with $p_i = \alpha_i$ or β_i. We denote the members of $\P_c^n$ by $p_i^*(s)$, $i=1,2,\cdots,2^n$.

We need the following lemma which follows from the convexity of $\P^n$.

Lemma 2.1

For a given Hurwitz polynomial $b(s)$ if $p_i^*(s)/b(s)$ is SPR for all $p_i^*(s) \varepsilon \P_c^n$ then $p(s)/b(s)$ is SPR $\forall p(s) \varepsilon \P^n$.

Proof

The following four polynomials are defined to be the Kharitonov polynomials [10].

$$K_1(s) = \alpha_n + \alpha_{n-1}s + \beta_{n-2}s^2 + \beta_{n-3}s^3 + \alpha_{n-4}s^4 + \alpha_{n-5}s^5 + \beta_{n-6}s^6 + \cdots + s^n$$

$$K_2(s) = \alpha_n + \beta_{n-1}s + \beta_{n-2}s^2 + \alpha_{n-3}s^3 + \alpha_{n-4}s^4 + \beta_{n-5}s^5 + \beta_{n-6}s^6 + \cdots + s^n$$

$$K_3(s) = \beta_n + \beta_{n-1}s + \alpha_{n-2}s^2 + \alpha_{n-3}s^3 + \beta_{n-4}s^4 + \beta_{n-5}s^5 + \alpha_{n-6}s^6 + \cdots + s^n$$

$$K_4(s) = \beta_n + \alpha_{n-1}s + \alpha_{n-2}s^2 + \beta_{n-3}s^3 + \beta_{n-4}s^4 + \alpha_{n-5}s^5 + \alpha_{n-6}s^6 + \cdots + s^n$$

Theorem 2.1 states the principle result of this section.

Theorem 2.1

Given $\P^n$ and a Hurwitz polynomial $b(s)$, $p(s)/b(s)$ is SPR $\forall p(s) \varepsilon \P^n$ iff $K_i(s)/b(s)$ is SPR $\forall i=1,2,3,4$.

∇∇∇

We use Lemmas 2.2 and 3.3 to prove this theorem. But first we express the Kharitonov polynomials as follows.

$$K_1(s) = h_1(s^2) + s\ g_1(s^2) \tag{2.1}$$
$$K_2(s) = h_1(s^2) + s\ g_2(s^2) \tag{2.2}$$
$$K_3(s) = h_1(s^2) + s\ g_2(s^2) \tag{2.3}$$
$$K_4(s) = h_1(s^2) + s\ g_1(s^2) \tag{2.4}$$

where

$$h_1(s^2) = \alpha_n + \beta_{n-2}s^2 + \alpha_{n-4}s^4 + \beta_{n-6}s^6 + \cdots\cdots \tag{2.5}$$

$$h_2(s^2) = \beta_n + \alpha_{n-2}s^2 + \beta_{n-4}s^4 + \alpha_{n-6}s^6 + \cdots\cdots \tag{2.6}$$

$$g_1(s^2) = \alpha_{n-1} + \beta_{n-3}s^2 + \alpha_{n-5}s^4 + \beta_{n-7}s^6 + \cdots\cdots \tag{2.7}$$

$$g_2(s^2) = \beta_{n-1} + \alpha_{n-3}s^2 + \beta_{n-5}s^4 + \alpha_{n-7}s^6 + \cdots\cdots \tag{2.8}$$

Note that for any $p(s) = h(s^2) + sg(s^2) \in \P^n$, $h_1(-\omega^2) \leq h(-\omega^2) \leq h_2(-\omega^2)$ and $g_1(-\omega^2) \leq g(-\omega^2) \leq g_2(-\omega^2)$.

Lemma 2.2

Consider any polynomial $p(s) \in \P_c{}^n$ with

$p(s) = h(s^2) + sg(s^2)$, and define

$p_1(s) = h_1(s^2) + sg(s^2) \in \P^n$

and

$p_2(s) = h_2(s^2) + sg(s^2) \in \P^n$.

with $h_1(s^2)$ and $h_2(s^2)$ defined in (2.5) and (2.6). Then if for a given $b(s)$, $\frac{p_1(s)}{b(s)}$ and $\frac{p_2(s)}{b(s)}$ are SPR then so is $\frac{p(s)}{b(s)}$.

Proof of Lemma 2.2

By hypothesis $\forall \omega \in R$, $i=1,2$, and with $\Delta(j\omega) = 1/b(j\omega)$

$$\mathrm{Re}[\{h_i(-\omega^2) + j\omega g(-\omega^2)\}\ \Delta(j\omega)] > 0 \tag{2.9}$$

Suppose with $p(s) = h(s^2) + sg(s^2) \in \P^n$, for some $\omega_1 \in R$

$$\mathrm{Re}[\{h(-\omega_1^2) + j\omega_1 g(-\omega_1^2)\}\ \Delta(j\omega)] \leq 0 \tag{2.10}$$

Then

$$\mathrm{Re}[\{h_1(-\omega^2) + j\omega_1 g(-\omega_1^2)\}\ \Delta(j\omega_1)] + \{h(-\omega_1^2) - h_1(-\omega_1^2)\}\mathrm{Re}\ [\Delta(j\omega_1)] \leq 0$$

Whence, since $h(-\omega_1^2) \geq h_1(-\omega_1^2)$, Re $[\Delta(j\omega_1)] < 0$. Thus,

$$\mathrm{Re}[\{h_2(-\omega_1^2) + j\omega_1 g(-\omega_1^2)\}\ \Delta(j\omega_1)]$$

$$= \mathrm{Re}[\{h_1(-\omega_1^2) + j\omega_1 g(-\omega_1^2)\}\ \Delta(j\omega_1)] + \{h_2(-\omega_1^2) - h_1(-\omega_1^2)\}\mathrm{Re}[\Delta(j\omega_1)] < 0.$$

Thus (2.9) is violated for i=2, and (2.10) cannot hold.

∇∇∇

Lemma 2.3

Consider any polynomial $q(s) \in \P_c^n$ with

$$q(s) = h(s^2) + sg(s^2) \text{ and}$$
$$q_1(s) = h(s^2) + sg_1(s^2) \in \P^n$$
$$q_2(s) = h(s^2) + sg_2(s^2) \in \P^n$$

with $g_1(s^2)$ and $g_2(s^2)$ defined in (2.7) and (2.8). Then if for a given $b(s)$, $\frac{q_1(s)}{b(s)}$ and $\frac{q_2(s)}{b(s)}$ are SPR then so is $\frac{q(s)}{b(s)}$.

Proof of Lemma 2.3 is similar to that of Lemma 2.2

∇∇∇

Proof of Theorem 2.1

The necessity is obvious as $K_i(s) \in \P^n \; \forall \; i=1,2,3,4$. For sufficiency note that any polynomial $p(s) \in \P_c^n$ can be written as

$$p(s) = \overline{h}(s^2) + s\overline{g}(s^2).$$

Now by Lemma 2.2, since $\{h_1(s^2) + sg_i(s^2)\}/b(s)$ and $\{h_2(s^2) + sg_i(s^2)\}/b(s)$ are SPR for $i = 1,2$, we get $\{\overline{h}(s^2) + sg_1(s^2)\}/b(s)$ and $\{\overline{h}(s^2) + sg_2(s^2)\}/b(s)$ are SPR.

Then by Lemma 2.3 $\overline{h}(s^2) + s\overline{g}(s^2)/b(s)$ is SPR, whence from Lemma 2.1 the theorem follows.

∇∇∇

3. CONDITION FOR FINDING b(s)

We now establish a "Routh like" condition on the coefficients α_i, β_i for which b(s) exists and show how, given the satisfaction of this condition b(s) can be constructed.

Consider any $a(s) \in \P^n$ given by

$$a(s) = a_0s^n + a_1s^{n-1} + \cdots + a_n; \qquad a_0=1 \tag{3.1}$$

and a b(s) of degree n

$$b(s) = b_0s^n + b_1s^{n-1} + \cdots + b_n \tag{3.2}$$

The following lemma shows that if $a(s)$ is Hurwitz and if $\mathrm{Re}[b(j\omega)/a(j\omega)] > 0 \ \forall \omega \varepsilon R$ then $b(s)$ must be Hurwitz.

Lemma 3.1

For any Hurwitz polynomial $a(s)$ if

$$\mathrm{Re}[b(j\omega)/a(j\omega)] > 0\text{-}\forall \omega \varepsilon R \tag{3.3}$$

for some polynomial $b(s)$, then $b(s)$ is Hurwitz.

Proof

Observe that (3.3) ensures that $b(s)$ has no pure imaginary zeros. Suppose it has a zero with positive real part. Then there exists $k>0$ such that $1 + k[b(s)/a(s)]$ has a zero in the right half plane of the complex domain. Also, since $a(s)$ is Hurwitz, $kb(j\omega)/a(j\omega)$ must encircle the $(-1,0)$ point at least once, whence there exists ω' such that $\mathrm{Re}\ [kb(j\omega')/a(j\omega')] < 0$ which violates (3.3). Hence, $b(s)$ is Hurwitz.

∇∇∇

Now since $a(s)$ is Hurwitz by definition, the transfer function $G(s) = a(s)/b(s)$ is SPR if

$$\mathrm{Re}\ [G(j\omega)] > 0 \ \forall \omega \varepsilon R$$

Now, sign of $\mathrm{Re}\ [a(j\omega)/b(j\omega)]$ is the same as that of $\mathrm{Re}\ [a(j\omega)b(-j\omega)]$.

$$a(j\omega)b(-j\omega) = \{ \sum_{i=0}^{n} a_i (j\omega)^{n-i} \} \{ \sum_{k=0}^{n} b_k (-j\omega)^{n-k} \}$$

Rearranging with $i+k=2\ell$ and $a_j = 0$ for $j < 0$ and $j > n$ we get

$$\mathrm{Re}\ [a(j\omega)b(-j\omega)] = \sum_{\ell=0}^{n} \{ \sum_{k=0}^{n} a_{2\ell-k}(-1)^{\ell+k} \} \ \omega^{2(n-\ell)} \tag{3.4}$$

Defining c_ℓ to be the coefficient of $\omega^{2(n-\ell)}$ in (3.4)

$$c_\ell = \sum_{k=0}^{n} a_{2\ell-k} b_k (-1)^{\ell+k}$$

Thus, if b is the $(n+1)$ vector with coefficients of $b(s)$ as its elements and c is the $(n+1)$ vector with coefficient of $\omega^{2(n-\ell)}$ in $\mathrm{Re}[a(j\omega)b(-j\omega)]$ as it's ℓ^{th} element, then we can write

$$c = Ab \tag{4.10}$$

with A as $(n+1) \times (n+1)$ matrix given by

$$a_{ij} = (-1)^{i+j} a_{2i-j}$$

where a_i are the coefficients of $a(s)$ and $a_k = 0 \ \forall k < 0$ and $\forall k > n$. Also, $a_0 = 1$. Now a sufficient conditions for $G(s) = a(s)/b(s)$ to be SPR is that $c_i > 0 \ \forall i \varepsilon \{0,1,\cdots,n\}$

We define a matrix A^* as

$$a^*_{ij} = \begin{cases} -\beta_{2i-j} & \text{if } i+j \text{ is odd} \\ \alpha_{2i-j} & \text{if } i+j \text{ is even} \end{cases}$$

In effect A^* is constructed by setting the negative elements of A to the corresponding β_i and the positive elements to corresponding α_i.

We then have the following lemma.

Lemma 3.2

A sufficient condition for a polynomial b(s) to exist such that a(s)/b(s) is SPR for all $a(s) \in \P^n$ is that there exist $b > 0$ and $c^* > 0$ such that

$$A^*b = c^* \tag{4.17}$$

Moreover, if b and c^* exist and with $b \triangleq [b_0, b_1, \cdots, b_n]^T$, the required polynomial is $b(s) = b_0 s^n + b_1 s^{n-1} + \cdots + b_n$.

Proof

If b and c^* exist such that

$A^*b > 0$ and $b > 0$

then, by the construction of A^*

$Ab > 0$

for every A corresponding to the polynomials in $\P^n$. Thus the result follows.

We now develop conditions on α_i and β_i such that $A^*b = c^*$ and $b, c^* > 0$. We construct two arrays $R^{(1)}$ and $R^{(2)}$, similar in structure and construction to the well known Routh array.

The first two rows of $R^{(1)}$ and $R^{(2)}$ are the first two rows of the Routh array corresponding to the Kharitonov polynomials $s^n + \alpha_1 s^{n-1} + \alpha_2 s^{n-2} + \beta_3 s^{n-3} + \beta_4 s^{n-4} + \alpha_5 s^{n-5} + \cdots$ and $s^n + \beta_1 s^{n-1} + \beta_2 s^{n-2} + \alpha_3 s^{n-3} + \alpha_4 s^{n-4} + \beta_5 s^{n-5} + \cdots$, respectively. Moreover, denoting the elements of the kth row of $R^{(i)}$ by $r^{(i)}_{kj}$, we have that for $k < n-1$

$$r^{(1)}_{k,j} = r^{(2)}_{k-2,j} - r^{(2)}_{k-1,j} \frac{r^{(2)}_{k-2,1}}{r^{(2)}_{k-1,1}}$$

$$r^{(2)}_{k,j} = r^{(1)}_{k-2,j} - r^{(1)}_{k-1,j} \frac{r^{(1)}_{k-2,1}}{r^{(1)}_{k-1,1}}$$

The nth rows are obtained as follows

If n is odd $r^{(1)}_{n,1} = r^{(2)}_{n-1,2}$ and $r^{(2)}_{n,1}$ is computed as above

If n is even $r^{(2)}_{n,1} = r^{(1)}_{n-1,2}$ and $r^{(1)}_{n,1}$ is computed as above

Thus, for example, for n = 4 the arrays $R^{(1)}$ and $R^{(2)}$ become:

$$R^{(1)} \qquad \begin{array}{lll} 1 & \alpha_2 & \beta_4 \\ \alpha_1 & \beta_3 & \\ \beta_2 - \dfrac{\alpha_3}{\beta_1}, & \alpha_4 & \\ \alpha_3 - \dfrac{\beta_4\beta_1}{\alpha_2 - \dfrac{\beta_3}{\alpha_1}} & & \end{array} \qquad\qquad R^{(2)} \qquad \begin{array}{lll} 1 & \beta_2 & \alpha_4 \\ \beta_1 & \alpha_3 & \\ \alpha_2 - \dfrac{\beta_3}{\alpha_1}, & \beta_4 & \\ \alpha_4 & & \end{array}$$

Then as the theorem below states, a sufficient condition for the existence of a b(s), whose ratio with all members of $\P^n$ is SPR, is that all elements in the first column of the two arrays, given above be positive. The proof of the theorem, which is omitted is constructive and yields a procedure for designing b(s), [16].

Theorem 3.1

For the set $\P^n$ defined before suppose every element in the first columns of $R^{(1)}$ and $R^{(2)}$ are positive. Then there exists an nth order polynomial b(s) such that b(s)/p(s) is SPR $\forall$ p(s) ε $\P^n$.

∇∇∇

4. CONCLUSION

We have addressed certain issues related to the question of designing SPR transfer functions. In particular we have shown that if a polynomial p(s) belongs to a set of Hurwitz polynomials defined by the bounds on the coefficients of p(s) then irrespective of the degree of p(s) there exist four polynomials $K_1(s) \cdots K_4(s)$ such that if $b(s)/K_i(s)$ is SPR for i=1, $\cdots$ 4, b(s)/p(s) is also SPR. Furthermore, a Routh type condition for the existence of such a b(s) is derived. A method for constructing b(s) is also given. The problem setting of this paper is primarily for output error adaptive identification. However, in view of the fact that SPR conditions are also linked to robust behaviour of adaptive controllers, our results, we feel, should also impact the design of robust adaptive controllers [15].

REFERENCES

[1] P.C. Parks, "Lyapunov redesign of model reference adaptive control schemes," *IEEE Transactions on Auto. Contr.*, AC-11, 3, pp. 362-367, 1966.

[2] Y.D. Landau, *Adaptive control: The model reference approach*, Marcel Dekker, Inc., 1979.

[3] G. Lüders and K.S. Narendra, "Stable adaptive schemes for estimation and identification of linear systems," *IEEE Trans. on Auto Contr.*, AC-19, pp. 841-847, Dec. 1974.

[4] A.P. Morgan and K.S. Narendra, "On the stability of non-autonomous differential equations $\dot{x}(t) = (A+B(t))x(t)$ with skew symmetric matrix $B(t)$," *SIAM J. Cont. Opt.*, 15, pp. 163-167, Jan. 1977.

[5] K.S. Narendra and L.S. Valavani, "A comparison of Lyapunov and hyperstability approaches to adaptive control of continuous systems," AC-25, pp. 243-247, April 1980.

[6] C.R. Johnson, Jr., "A convergence proof for a hyperstable adaptive recursive filter," *IEEE Trans. on Info. Thy.*, IT-25,, pp. 746-749, Nov. 1979.

[7] B.D.O. Anderson, "An approach to multivariable system identification," *Automatica*, 13, pp. 401-408, 1977.

[8] B.D.O. Anderson, "Exponential stability of linear equations arising in adaptive identification," *IEEE Trans. on Auto Contr.*, AC-22, pp. 83-88, Feb. 1977.

[9] S. Dasgupta, B.D.O. Anderson and R.J. Kaye, "Output error identification methods for partially known systems," *Int. J. Control*, 43, 1, pp. 177-191, 1986.

[10] V.I. Kharitonov, "Asymptotic stability of an equilibrium position of a family of systems of linear differential equations," *Differential Equations*, pp. 1483-1485, Nov. 1985.

[11] F.R. Gantmacher, *Theory of Matrices*, Vol. II, Chelsea Publishers.

[12] R.R. Bitmead, "Toeplitz matrices and explicit solutions of the discrete time Lyapunov matrix equations," *Technical Report*, James Cook University of North Queensland, Australia, May 1980.

[13] E.I. Jury, *Inners and Stability of Dynamic Systems*, Chapter 5, Krieger Publishing Co., 1982.

[14] D.G. Luenberger, *Optimization by Vector Space Methods*, John Wiley and Sons, 1969.

[15] B.D.O. Anderson, R.R. Bitmead, C.R. Johnson Jr., P.V. Kokotovic, R.L. Kosut, L. Mareels, L. Praly and B. Riedle, *Stability of Adaptive Systems: Passivity and Averaging Analysis*, MIT Press, in press.

[16] S. Dasgupta and A.S. Bhagwat, "Conditions for designing strictly positive real transfer functions for adaptive output error identification," submitted for publication.

REMARKS ON COMPLEXITY OF SYSTEM IDENTIFICATION

GLENN K. HEITMAN and WILLIAM L. ROOT
Aerospace Engineering Department
The University of Michigan
Ann Arbor, Michigan 48109-2140

ABSTRACT

This paper considers two ways of describing identification complexity in a certain system formulation, possibly as first steps toward a suitable definition. These are: (i) the number of tests that must be performed and (ii) the size of the class of identification models as measured by its metric entropy when the entropy exists. In regard to the first of these, a theorem about interpolative representations is proved, and a bound on the number of tests is obtained when the system mappings satisfy a Lipschitz condition. In regard to the second, estimates of the metric entropy of some classes of Lipschitz mappings are obtained.

INTRODUCTION

This paper is a very preliminary study of the complexity of identification of abstract input-output systems. The systems are modeled as (generally nonlinear) mappings from a metric input space U to a normed linear output space Y. Identification, as it is to be understood here, means the following: given U and Y, a mathematical system model (called the identification model) is to be determined such that for any $u \in U$ the corresponding output given by the identification model is to be the same, or more usually approximately the same, as that produced by the system. The identification model is to be determined from a *finite* set of input-output tests with the system in the same condition for each test; i.e., the input-output map of the system is to be the same for each test. This situation can obviously be obtained if the unknown system belongs to a class of time-invariant dynamical systems with bounded memory; otherwise one must resort to approximations with uniformly decaying memories or take account of system state, neither of which is done here (note, however, that even with the assumption of finite memory, tests can overlap in time; see [1]). It is assumed throughout that tests are performed with noiseless observations.

No single definition is proposed here for the term "identification complexity"; rather, what is done is to consider two ways of describing some aspects of identification complexity as preliminary steps toward a suitable definition. These are: (1) the number of tests that must be performed to achieve identification to within some specified tolerance, and (2) some measure of the number of identification models one has to choose from; i.e., a measure of the size of the class of unknown systems.

TECHNICAL BACKGROUND

The relevant notations and definitions concerning compound systems and ϵ-representations (see [2]) are collected in Appendix 1. Prop. 1.7 of [2] states the following result. Given a compound system $S = (Y, f, X, U)$, let U and X be compact. (Then in the natural representation $\overline{S} = (Y, \overline{f}, \mathbf{H}, U)$, $\mathbf{H}$ is also compact.) Under these conditions there always exists an ϵ-representation (S', ϕ) possessing the following properties: X' is a subset of a finite-dimensional normed linear space; ϕ is continuous; ϕ depends only on $F \in \mathbf{H}$ restricted to a finite set of elements in U (this set $\{\overline{u}_1, \ldots, \overline{u}_N\}$ is called the *test set*); ψ' is a restriction of a linear map, and so is ϕ if Y is a Hilbert space. The proof is by construction, and in the case where Y is finite-dimensional the ϵ-representation is given by

$$\hat{F}(u) \equiv [(\psi' \cdot \phi)(F)](u) = \sum_{n=1}^{N} \gamma_n(u) F(\bar{u}_n) \tag{1}$$

where the real-valued interpolation functionals γ_n satisfy the following properties: (i) $0 \leq \gamma_n(u) \leq 1$ for $u \in U$ and $n = 1, \ldots, N$; (ii) $\gamma_n(\bar{u}_k) = \delta_{nk}$; (iii) $\gamma_n(u) = 0$ if $d((u, \bar{u}_n) \geq \eta$ for some fixed η; (iv) $\sum_{n=1}^{N} \gamma_n(u) = 1$ for $u \in U$; (v) γ_n is continuous on U. (See [2],[3].)

An ϵ-representation is obviously an approximate identification with guaranteed accuracy (in the noise-free case), and if it is of the form of Eq.(1), the number of tests is given to be N. Thus N is an upper bound on the number of tests that must be performed to obtain an ϵ-identification. An identification complexity problem is then: under specified conditions on U, Y, and $\mathbf{H}$ and for given $\epsilon > 0$, to obtain estimates of the smallest possible values of N. Note that for *any* representation of the form of Eq.(1) the parameter space X' is a subset of the product space Y^N (it may be all of Y^N). This is true whether Y is finite-dimensional or not.

The other notion having to do with identification complexity previously mentioned is to get some kind of measure of the size of the space $\mathbf{H}$. For those cases where $\mathbf{H}$ is totally bounded it seems appropriate to obtain estimates of ϵ-entropy (or metric entropy), which is defined as follows (see [4]). Let M be a metric space and V a set in M. A system $\{A_\alpha\}$ of sets in M is an *ϵ-covering* of V if $dia(A_\alpha) \leq 2\epsilon$ and $V \subset \bigcup_\alpha A_\alpha$. If V is totally bounded, for any $\epsilon > 0$ there exists a finite ϵ-covering. Define $N_\epsilon(V)$ to be the minimum number of sets in an ϵ-covering of V; then $H_\epsilon(V) = \log_2 N_\epsilon(V)$ is the *ϵ-entropy* of V.

A related notion is defined in terms of ϵ-nets (a set $\{x_1, \ldots, x_p\} \subset M$ is a (finite) ϵ-net for V if every $x \in V$ satisfies $d(x, x_k) \leq \epsilon$ for some $k = 1, \ldots, p$; if V is totally bounded it has a finite ϵ-net). Let $P_\epsilon(V)$ be the minimum number of elements in any ϵ-net for V, then the following elementary inequalities are satisfied.

$$N_\epsilon(V) \leq P_\epsilon(V) \leq N_{\epsilon|2}(V) \quad .$$

If the space M is centered (see the definition after Prop. 4) $N_\epsilon(V) = P_\epsilon(V)$.

RESULTS

The first result effectively generalizes Prop. 1.7 of [2] referred to above by removing the requirement that U be a compact space.

Proposition 1

Given a compound system $\bar{S} = (Y, \bar{f}, \mathbf{H}, U)$ in natural form with U a connected metric space, $\mathbf{H}$ totally bounded in $\mathbf{C}^\infty(U, Y)$, and $\mathbf{H}(U)$ totally bounded in Y, then for each $\epsilon > 0$ there exists an ϵ-representation of the form of Eq.(1). If Y is finite-dimensional, the ϵ-representation is finitely parametrized.

Proof: See Appendix 2.

Remark: The condition that $\mathbf{H}(U)$ be totally bounded is not an additional requirement, since it was implied by the hypotheses of the earlier theorem.

The next result is set in the context of the earlier ϵ-representation theorem (Prop. 1.7 of [2]) where U is required to be totally bounded. The class $\mathbf{H}$ chosen is not totally bounded in

$\mathbf{C}^\infty$, but still an interpolative ϵ-representation is given as in Eq. (1) and furthermore the number of test inputs is obtained.

Proposition 2

Let U be a totally bounded metric space and Y a finite-dimensional Banach space. Let $\mathbf{H} \subset \mathbf{C}^\infty(U,Y)$ be the class of Lipschitz mappings with Lipschitz constant not exceeding K. Then for any $\epsilon > 0$, an interpolative ϵ-representation is given an in Eq.(1) with the test set $\{\bar{u}_1, \ldots, \bar{u}_N\}$ chosen to be an $\alpha\delta$-net in U, where $\delta = \epsilon/3K$ and α is any positive number less than 1. The N in the representation is $N = P_{\alpha\delta}(U)$.

Proof: Interpolation functionals can be constructed with all the properties listed above with $\eta = \delta$; i.e., $\gamma_n(u) = 0$ if $d(u,u_n) \geq \delta$. For any $\bar{u}_n$,

$$\begin{aligned} |||\, F(u) - \hat{F}(u) \,||| \leq |||\, F(u) - F(\bar{u}_n) \,||| &+ |||\, F(\bar{u}_n) - \hat{F}(\bar{u}_n) \,||| \\ &+ |||\, \hat{F}(\bar{u}_n) - \hat{F}(u) \,||| \end{aligned} \tag{*}$$

We choose $\bar{u}_n$ so that $d(u,\bar{u}_n) \leq \delta$. Then $|||\, F(u) - F(\bar{u}_n) \,||| \leq Kd(u,\bar{u}_n) = K\delta = \epsilon/3$. The second term on the right in the inequality (*) is zero. To bound the third term we let $\{\bar{u}_{j_1}, \ldots, \bar{u}_{j_P}\}$ be those $\bar{u}_n$ within δ of u. Then

$$\begin{aligned} |||\, \hat{F}(\bar{u}_n) - \hat{F}(u) \,||| &= |||\, \hat{F}(\bar{u}_n) - \sum_{k=1}^{N} \gamma_k(u) F(\bar{u}_k) \,||| \\ &= |||\, \sum_{k=1}^{N} \gamma_k(u) \,[\, \hat{F}(\bar{u}_n) - F(\bar{u}_k) \,]\, ||| \\ &\leq \sum_{p=1}^{P} \gamma_{j_p}(u)\, |||\, \hat{F}(\bar{u}_n) - F(\bar{u}_{j_p}) \,||| \\ &\leq \sum_{p=1}^{P} \gamma_{j_p}(u)\, [\, |||\, F(\bar{u}_n) - F(u) \,||| + |||\, F(u) - F(\bar{u}_{j_p}) \,|||\,] \\ &\leq K\delta + K\delta = 2\epsilon/3 \quad . \end{aligned}$$

Hence, $|||\, F(u) - \hat{F}(u) \,||| \leq \epsilon$ which implies that $||\, F - \hat{F} \,||_\infty \leq \epsilon$. ■

Hence in the case of a totally bounded input space, Lipschitz mappings, and the interpolative identification model, one obtains a measure of the identification complexity (in the sense of number of tests) in terms of the identification tolerance ϵ and the Lipschitz constant K, namely $N = P_{\frac{\epsilon}{3K}}(U)$.

Example: M^{th}-degree polynomial integral operators. Let A and T be subsets of $\mathbb{R}^1$ (usually intervals). An M^{th}-degree Volterra polynomial is a mapping F of the form

$$[\, F(u) \,]\,(t) = y(t) = \sum_{m=0}^{M} \int_A \cdots \int_A x_m(t,s_1, \ldots, s_m) u(s_1) \cdots u(s_m) ds_1 \cdots ds_m \;, \quad t \in T \quad .$$

If $u \in L_2(A)$ and $x_m \in L_2(T \times A^m)$ for all m, then $y \in L_2(T)$. The following inequality from [2] will be needed:

$$|||\, F(u_1) - F(u_2) \,||| \leq ||\, u_1 - u_2 \,|| \cdot \sum_{m=1}^{M} m \,||\, x_m \,||\, [\, \max(||\, u_1 \,||, ||\, u_2 \,||) \,]^{m-1} \tag{2}$$

Now if U is totally bounded in $L_2(A)$, then (2) can be written

$$|||F(u_1) - F(u_2)||| \leq \| u_1 - u_2 \| \cdot \sum_{m=1}^{M} m \, \| x_m \| \; [\, \max\{ \| u \| : u \in U \} \,]^{m-1}$$

Furthermore, if the kernels are bounded in L_2-norm, say $\| x_m \| \leq C_m < \infty$, $m = 1, \ldots, M$, then

$$|||F(u_1) - F(u_2)||| \leq \| u_1 - u_2 \| \cdot \sum_{m=1}^{M} mC_m \; [\, \max(\| u \| : u \in U \} \,]^{m-1}$$

Hence, under these conditions on the input space and the kernels, the Volterra polynomials define Lipschitz mappings with Lipschitz constant not exceeding

$$K = \sum_{m=1}^{M} mC_m \; [\, \max\{ \| u \| : u \in U \} \,]^{m-1}$$

and so Prop. 2 applies; i.e., an interpolative representation is given as in Eq.(1). Note that in this example the parameter space X in the original system is a subset of $\sum_{m=0}^{M} \dot{+} L_2(T \times A^m)$, where $x = (x_0, x_1, \ldots, x_M)$ and $\| x \|^2 = \sum_{m=0}^{M} \| x_m \|^2$ ($\dot{+}$ denotes direct sum).

As mentioned previously, another way of approaching identification complexity is to estimate the ϵ-entropy of the class $\mathbf{H}$. (In [5] Zames calculates bounds on the ϵ-entropy of a class of transfer functions for linear systems.) Here, bounds are obtained on the ϵ-entropy of classes of Lipschitz mappings. These results follow essentially as corollaries of results in [4] for the case of real Lipschitz functionals on a metric space.

Proposition 3

Let U be a totally bounded metric space, and Y a Banach space. $\mathbf{H}$ is the class of Lipschitz mappings from U into Y with Lipschitz constant not exceeding K. Then

$$N_\epsilon(\mathbf{H}) \leq P_\epsilon(\mathbf{H}) \leq [\, N_{\epsilon/4}(\mathbf{H}(U)) \,]^{N_{\frac{\epsilon}{4K}}(U)}$$

or $\log_2 H_\epsilon(\mathbf{H}) \leq H_{\frac{\epsilon}{4K}}(U) + \log \log H_{\frac{\epsilon}{4}}(\mathbf{H}(U))$.

Proof: Let $V_1, \ldots, V_N$, $N = N_{\frac{\epsilon}{4K}}(U)$ be an $\frac{\epsilon}{4K}$-covering of U, and let $W_1, \ldots, W_L$, $L = N_{\frac{\epsilon}{4}}(\mathbf{H}(U))$ be an $\frac{\epsilon}{4}$-covering of $\mathbf{H}(U)$. Choose points $\{u_1, \ldots, u_N\}$, with $u_i \in V_i$. Given $F \in \mathbf{H}$, $F(u_i) \in W_j$ for some j, for $i = 1, \ldots, N$. Hence $|||F(u_i) - y_j||| \leq \epsilon/2$, where $y_1, \ldots, y_L$ are points in $W_1, \ldots, W_L$. On V_i set $\widetilde{F}(u) = y_j$, $i = 1, \ldots, N$. Then

$$|||F(u) - \widetilde{F}(u)||| \leq |||F(u) - F(u_i)||| + |||F(u_i) - \widetilde{F}(u)|||$$

$$\leq K \cdot \frac{\epsilon}{2K} + \frac{\epsilon}{2} = \epsilon \quad .$$

Hence, $\| F - \widetilde{F} \|_\infty \leq \epsilon$; i.e., the $\widetilde{F}$'s form an ϵ-net for $\mathbf{H}$. On each V_i, $\widetilde{F}$ can assume no more than L values, and so there are no more than L^N such $\widetilde{F}$'s. ■

If Y is restricted to be finite-dimensional, then the following two results are obtained.

Proposition 4

Let U be a totally bounded metric space and let Y be d-dimensional space with $|||\, y \,||| = \max(\,|y_1|, \ldots, |y_d|\,)$, $y = (y_1, \ldots, y_d)$. Let $\mathbf{H}$ be the class of bounded Lipschitz mappings from U into Y with Lipschitz constant not exceeding K and bound C. Then

$$N_\epsilon(\mathbf{H}) \le P_\epsilon(\mathbf{H}) \le \left(2\left\lfloor \frac{2C}{\epsilon} \right\rfloor + 1\right)^{dN_{\frac{\epsilon}{4K}}(U)} \quad . \tag{3}$$

Proof: Cover U by $V_1, \ldots, V_N$, $N = N_{\frac{\epsilon}{4K}}(U)$, and choose points $u_i \in V_i$. Write $F(u)$ as $(F_1(u), \ldots, F_d(u))$. Since F is Lipschitz (K) and bounded by C, the same is true of $F_1, \ldots, F_d$. For each F_m, $m = 1, \ldots, d$, define the approximation $\widetilde{F}_m$ by

$$\widetilde{F}_m(u) = \left\lfloor \frac{2F_m(u_i)}{\epsilon} \right\rfloor \frac{\epsilon}{2} \; , \quad u \in V_i \quad .$$

This approximates $F_m(u_i)$ to within $\epsilon/2$ on V_i. Then put $\widetilde{F}(u) = (\widetilde{F}_1(u), \ldots, \widetilde{F}_d(u))$, $u \in U$. Then

$$\begin{aligned} |||\, F(u) - \widetilde{F}(u) \,||| &\le |||\, F(u) - F(u_i) \,||| + |||\, F(u_i) - \widetilde{F}(u) \,||| \\ &= \max\Big(\, |F_1(u) - F_1(u_i)|, \ldots, |F_d(u) - F_d(u_i)| \,\Big) \\ &\quad + \max\Big(\, |F_1(u_i) - \widetilde{F}_1(u)|, \ldots, |F_d(u_i) - \widetilde{F}_d(u)| \,\Big) \\ &\le \epsilon/2 + \epsilon/2 = \epsilon \quad . \end{aligned}$$

Hence, the $\widetilde{F}$'s form an ϵ-net for $\mathbf{H}$. $\widetilde{F}$ assumes no more than $\left(2\left\lfloor \frac{2C}{\epsilon} \right\rfloor + 1\right)^d$ values on each V_i, and so the conclusion follows. ■

A metric space M is said to be *centered* if in every subset A of diameter δ, there is a point a' such that every $a \in A$ is within distince $\delta/2$ of a'. If the input space is required to be centered and connected, then a tighter bound on $N_\epsilon(\mathbf{H})$ is obtained.

Proposition 5

Let U, $\mathbf{H}$, and Y be as in Prop. 4. In addition, assume that U is connected and centered. Then

$$N_\epsilon(\mathbf{H}) \le \left(2\left\lfloor \frac{2C}{\epsilon} \right\rfloor + 1\right)^d \, 2^{dN_{\frac{\epsilon}{4K}}(U)} \quad . \tag{4}$$

Proof: Again, let $V_1, \ldots, V_N$, $N = N_{\frac{\epsilon}{4K}}(U)$, cover U. Now, choose $u_1, \ldots, u_N$ to be *centers* of $V_1, \ldots, V_N$. On V_1, approximate each F_m, $m = 1, \ldots, d$ by $\widetilde{F}_m(n) = \left\lfloor \frac{2F(u_1)}{\epsilon} \right\rfloor \epsilon/2$, then put $\widetilde{F}(u) = (\widetilde{F}_1(u), \ldots, \widetilde{F}_d(u))$, $u \in V_1$. Consider those V_i which meet V_1; the values of F at the centers u_i satisfy $|||\, F(u_i) - \widetilde{F}(u_1) \,||| \le |||\, F(u_i) - F(u_1) \,||| + |||\, F(u_1) - \widetilde{F}(u_1) \,||| \le \epsilon/2 + \epsilon/2 = \epsilon$. In view of the connectedness of U, one can construct the entire $\widetilde{F}$, going from center to center, choosing one of the 2^d

values $V_1(u_i) \pm \epsilon/2, \ldots, F_d(u_i) \pm \epsilon/2$ at each step. Since there are $\left(2\left\lfloor \frac{2C}{\epsilon} \right\rfloor + 1\right)^d$ $2^{dN_{\frac{\epsilon}{4K}}(U)}$ such $\widetilde{F}$'s, the proof is complete. ■

Remark: It is easy to see that (4) is indeed a tighter bound than (3).

CONCLUDING REMARKS

As indicated in the Introduction, this paper presents a preliminary study on the subject of identification complexity. There are several questions that we intend to pursue further. For example, Proposition 1 shows that an interpolative ϵ-identification can be obtained with a finite number of tests even if U is unbounded. In this case, the number of tests is still a useful way to describe the identification complexity, but it cannot be related to the metric entropy of U since the entropy does not exist. Similarly, it is of interest to consider the case where $\mathbf{H}$ is not totally bounded (in particular where $\mathbf{H}$ is a linear space) and hence again the metric entropy does not exist. So the question of what is a suitable general measure of identification complexity needs to be addressed. Also, it has been assumed that identification is achieved using a sequence of tests, with the system in the same condition for each test. It is of course of interest to consider the case of testing a causal system with a single continuing input and to relate the identification complexity to the *time* required to achieve identification.

APPENDIX 1

Some notations and definitions concerning compound systmes and ϵ-representations are summarized here (see [2]).

Let X be a space of parameters determining all admissible input-output mappings from U into Y. Let $f : X \times U \to Y$ be a mapping, then $S = (Y, f, X, U)$ is a *compound system*. For what follows it is required that X and U be metric spaces and Y be a Banach space. $\mathbf{C}^\infty(U,Y)$ denotes the Banach space of continuous mappings with bounded range from U into Y with the uniform norm

$$\| F \|_\infty = \sup_{u \in U} ||| F(u) |||$$

($||| \cdot |||$ denotes the norm in Y). It is required that f be continuous on $X \times U$, and that each $f(x,\cdot)$ be bounded. Let $\mathbf{H}$ be the set of all mappings $F : U \to Y$ defined by $F(\cdot) = f(x,\cdot)$, $x \in X$. By the conditions imposed on f, $\mathbf{H} \subset \mathbf{C}^\infty(U,Y)$. Define $\overline{f} : \mathbf{H} \times U \to Y$ by $\overline{f}(F,u) = F(u)$. Then $\overline{S}(Y, \overline{f}, \mathbf{H}, U)$ is a compound system called the *natural representation* of S. The mapping $\psi : X \to \mathbf{H}$ defined by $\psi(x) = F = f(x,\cdot)$ is called the *natural mapping* of S; ψ is continuous iff the functions $f(\cdot,u)$ are equicontinuous for all $u \in U$.

One kind of approximate system representation is defined in [2]: given a compound system $\overline{S} = (Y, \overline{f}, \mathbf{H}, U)$ in natural form, an *ϵ-representation* of $\overline{S}$ is a pair (S', ϕ), where $S' = (Y, f', X', U)$ is a compound system with natural mapping ψ', and $\phi : \mathbf{H} \to X'$ satisfies

$$\| F - (\psi' \cdot \phi)(F) \|_\infty \leq \epsilon \ , \quad F \in \mathbf{H} \ .$$

It is not required that $(\psi' \cdot \phi)(F)$ belong to $\mathbf{H}$.

APPENDIX 2

Proof of Proposition 1: Fix $\epsilon > 0$. Let $\{F_1, \ldots, F_K\}$ be an $\epsilon/4$-net for $\mathbf{H}^-$ (i.e., for any $F \in \mathbf{H}$ there is a k such that $\| F - F_k \|_\infty \leq \epsilon/4$; $\mathbf{H}^-$ denotes the closure of $\mathbf{H}$), and let $\{B_1, \ldots, B_L\}$ be open $\epsilon/4$-balls covering $\mathbf{H}(U)^-$. Put $\widetilde{B}_l^k = F_k^{-1}(B_l)$, $l = 1, \ldots, L$, $k = 1, \ldots, K$. Each $\widetilde{B}_l^k$ is open in U since F_k is continuous. Consider the nonempty sets $\widetilde{B}_{i_1}^1 \cap \widetilde{B}_{i_2}^2 \cap \cdots \cap \widetilde{B}_{i_K}^K$, $i_1, \ldots, i_K = 1, \ldots, L$. These sets are open, and they cover U: for each k, any $u \in U$ is in $F_k^{-1}(B_l) = \widetilde{B}_l^k$ for some l. Enumerate the nonempty $D_n = \widetilde{B}_{i_1}^1 \cap \ldots \cap \widetilde{B}_{i_K}^K$, $n = 1, \ldots, N$ in such a way that no D_n is contained in a union of previous ones; if a candidate for D_n is contained in the union of $D_1, \ldots, D_{n-1}$, discard it.

Now choose $\bar{u}_1 \in D_1$. Since U is connected, the only open/closed sets are ϕ and U. D_1 being open is therefore not closed, hence D_1 cannot be finite. So choose $\bar{u}_2 \in D_2$, $\bar{u}_2 \neq \bar{u}_1$; $\bar{u}_3 \in D_3$ distinct from $\bar{u}_1$ and $\bar{u}_2$; etc. In this way a set of distinct points $\{\bar{u}_1, \ldots, \bar{u}_N\}$ is obtained with $\bar{u}_n \in D_n$. Put $C_n = D_n \setminus \bigcup_{i \neq n} \{\bar{u}_i\}$; $n = 1, \ldots, N$. The C_n's are open and cover U.

For $u \in U$ define

$$f_n(u) = \frac{d(u, C_n^c)}{d(u, \bar{u}_n) + d(u, C_n^c)} \quad , \quad n = 1, \ldots, N \quad .$$

This is well-defined because $d(u, \bar{u}_n) + d(u, C_n^c) > 0$ for all u. Now define

$$\gamma_n(u) = \frac{f_n(u)}{\sum_{i=1}^{N} f_i(u)} \quad , \quad n = 1, \ldots, N \quad .$$

It is easy to see that the γ_n satisfy the following properties: (i) $\gamma_n(u) \geq 0$ for all n and u, (ii) $\sum_{n=1}^{N} \gamma_n(u) = 1$ for all u, (iii) $u \in C_n^c$ implies $\gamma_n(u) = 0$, (iv) $u = \bar{u}_n$ implies $\gamma_n(u) = 1$, and (v) γ_n is continuous on U.

Define the approximation

$$\hat{F}(u) = \sum_{n=1}^{N} \gamma_n(u) F(\bar{u}_n) \quad , \quad u \in U \quad .$$

It remains to be shown that $\hat{F}$ is an ϵ-approximation to F; i.e., $\| F - \hat{F} \|_\infty \leq \epsilon$. First, note that for any $k \in \{1, \ldots, K\}$,

$$\begin{aligned} ||| F(u) - \hat{F}(u) ||| \leq\ & ||| F(u) - F_k(u) ||| + ||| F_k(u) - \hat{F}_k(u) ||| \\ & + ||| \hat{F}_k(u) - \hat{F}(u) ||| \quad . \end{aligned}$$

Consider each of these terms:

1) Choose k such that $\| F - F_k \| \leq \epsilon/4$; then $||| F(u) - F_k(u) ||| \leq \epsilon/4$.

$$2)\quad |||\, F_k(u) - \hat{F}_k(u) \,||| = |||\, F_k(u) - \sum_{i=1}^{N} \gamma_n(u) F_k(\bar{u}_i) \,|||$$

$$= |||\, \sum_{i=1}^{N} \gamma_i(n) \Big[F_k(u) - F_k(\bar{u}_i) \Big] \,||| \qquad \textbf{(A2.1)}$$

$$\leq \sum_{i=1}^{N} \gamma_i(n) \, |||\, F_k(u) - F_k(\bar{u}_i) \,|||$$

Let $C_{n_1}, \ldots, C_{n_P}$ $(P \leq N)$ be the C_n's with $u \in C_n$. Then (A2.1) becomes

$$\sum_{p=1}^{P} \gamma_{n_p}(n) \, |||\, F_k(u) - F_k(\bar{u}_{n_p}) \,|||$$

because $\gamma_i(u) = 0$ for the other indices. But $|||\, F_k(u) - F_k(\bar{u}_{n_p} \,||| \leq \epsilon/2$ since $u \in C_{n_p}$. So finally,

$$|||\, F_k(u) - \hat{F}_k(u) \,||| \leq \epsilon/2 \quad .$$

$$3)\quad |||\, \hat{F}_k(u) - \hat{F}(u) \,||| = |||\, \sum_{n=1}^{N} \gamma_n(u) \Big[F_k(\bar{u}_n) - F(\bar{u}_n) \Big] \,|||$$

$$\leq \sum_{n=1}^{N} \gamma_n(n) \, |||\, F_k(\bar{u}_n) - F(\bar{u}_n) \,|||$$

$$\leq \| F_k - F \|_\infty \leq \frac{\epsilon}{4} \quad .$$

Hence, $\| F - \hat{F} \|_\infty \leq \epsilon/4 + \epsilon/2 + \epsilon/4 = \epsilon$. This completes the proof.

REFERENCES

[1] W.L. Root, "A Procedure for Identifying On-Line Nonlinear Systems Known to Belong to a Specified Compact Class" *Proc. of* 17^{th} *Annual Conf. on Info. Sciences and Systems,* The Johns Hopkins Univ. pp.276-281 (1983).

[2] W.L. Root, "On the Modeling of Systems for Identification. Part 1: ϵ-Representations of Classes of Systems" *SIAM J. Control,* 13n4, pp.945-974 (1975).

[3] W.L. Root, "On the Identifiability of Abstract Systems" *Proc. of* 19^{th} *Annual Conf. on Info. Sciences and Systems,* The Johns Hopkins Univ. pp.50-55 (1985).

[4] A.N. Kolmogorov and V.M. Tihomirov, "ϵ-Entropy and ϵ-Capacity of Sets in Functional Spaces" *Uspehi Mat. Nauk,* 14n2, pp.3-86 (1959). English Transl.: *Amer. Math. Soc. Translations,* 17, 2, pp.277-364 (1961).

[5] G. Zames, "On the Metric Complexity of Causal LInear Systems: ϵ-Entropy and ϵ-Dimension for Continuous Time" *IEEE Trans. on Automatic Control,* AC-24, no.2, pp.222-230 (1979).

ROBUST ADAPTIVE CONTROLLER

K. HAMADA , M. SUZUKI , and Y. ANDO
NAGOYA UNIVERSITY
FUROU-CHOU / CHIKUSA-KU / NAGOYA / JAPAN 464

ABSTRACT

A new modified adaptive algorithm with nonlinear decay term is proposed to guarantee the robust global stability of continuous-time model reference adaptive control systems to unmodeled dynamics in the plant. A sufficient condition to find the design parameter in the decay term which insures the robust result is proved and its dependence on some information of the tuned gain is discussed. Also the drawbacks of our new algorithm is discussed with a simple example.

I . INTRODUCTION

In general, global stability properties of model reference adaptive control (MRAC) systems are guaranteed under the “matching assumption” that the model order is not lower than the order of the unknown plant. But unfortunately this restrictive assumption is likely to be violated in applications (e.g.,by the existence of unmodeled dynamics in the plant). Several attempts have been made to formulate and analyze the robustness of adaptive schemes to such unmodeled dynamics . In paticular Ioannou and Kokotovic ([1],[2]) used the modified adaptive algorithm with linear decay term and have proved that the existence of aregion of attraction from which all signals converge to a residual set.

In this paper, we propose a new modified adaptive algorithm with nonlinear decay term and investigate the possibility that the MRAC system which uses our new algorithm will have the robust global stability properties to the unmodeled dynamics in the plant. The fundamental formulation and procedure of the analysis is the same which used in [3] and the concept of error system plays the main role.

II . PROBLEM FORMULATION

A. Notation

The input-output formulation of multivariable system is taken through out this paper and the notation and terminology used is standard (see, e.g.,[4]). About LTI systems we introduce the following notation. Let R(s) and $R_0(s)$ denote the proper and strictly proper rational functions, respectively. Let S(s) and $S_0(s)$ denote functions in R(s) and $R_0(s)$, respectively, whose poles all have negative real parts. Denote multi-variable systems with transfer function matrices by $R(s)^{n\times m}$, $S^{n\times m}$, etc. For example, $G\in S_0^{n\times m}$ mean that all elements of G belong to S_0, and so on.

When G is strictly positive real(SPR), denote $G\in SPR^m$. We define the following two subsets:

$$SPR^m{}_+ =\{G\in S^{m\times m} \mid \underline{\lambda}(1/2[G(j\omega)+G(-j\omega)^T]-\mu I)>0,\ {}^\forall\omega\in R\} \quad (2.1a)$$

$$SPR_0^m = \{G \in S_0^{m \times m} \mid \underline{\lambda}(1/2[G(j\omega)+G(-j\omega)^T] - \mu G(-j\omega)^T G(j\omega)) > 0, \ \forall \omega \in R\} \quad (2.1b)$$

where $\underline{\lambda}(A)$ denotes the smallest eigenvalue of A.

These notation are the same which used in [3] and see it in detail.

B. Structure of the Adaptive System

We consider the model reference adaptive control (MRAC) depicted in Fig.1, consisting of an uncertain plant P, a reference model H_r , and an adaptive controller $C(\hat{\theta})$, where $\hat{\theta} \in R^k$ is the adaptive gain vector, r is a reference input, d is a disturbance process, and n is sensor noise. The system of Fig.1 can be translated to the closed-loop form, such that

$$e = H(\hat{\theta})w \quad (2.2)$$

where $e(t) \in R^m$ is the error signal to be controlled, and $w(t)=(r, d, n)$ is the external input restricted to some set W. The class of adaptive controller considered here is such that the adaptive gains multiply elements of internal signal $z(t) \in R^k$, referred to as the regressor signals, to produce the adaptive control signals

$$u_i = \hat{\theta}_i z_i, \qquad i \in [1, m] \quad (2.3)$$

where $\hat{\theta}_i$ and z_i are k_i -dimensional subsets of the elements in $\hat{\theta}$ and z, respectively. Thus,

$$k = \sum_i k_i . \qquad i \in [1, m] \quad (2.4)$$

System (2.2),(2.3) can also be described in an error system form (e.g., [3],[5]) as follows.

Define the adaptive gain error

$$\theta(t) = \hat{\theta}(t) - \theta_* \quad (2.5)$$

where $\theta_* \in R^k$ is the constant vector of the tuned gain which would be selected if the plant P were known. Using (2.5), we can rewrite (2.3) as

$$u = Z^T \theta_* + v , \qquad v = Z^T \theta \quad (2.6)$$

where v is the adaptive control error, and Z is the time-varying matrix defined by Z = block diag $(z_1, z_2, \ldots, z_m)$.

The interconection system H_I is given to describe the relations among the signals e, z, v, and w. In particular, let $H_I \in R(s)^{(m+k) \times (m+q)}$, where H_I is defined by

$$\begin{bmatrix} e \\ z \end{bmatrix} = H_I \begin{bmatrix} w \\ v \end{bmatrix} = \begin{bmatrix} H_{ew} & -H_{ev} \\ H_{zw} & -H_{zv} \end{bmatrix} \begin{bmatrix} w \\ v \end{bmatrix} \quad (2.7)$$

The effect of this structure is to isolate the adaptive control error v from the rest of the system. When the adaptive control is tuned, $\theta = 0$

and $v = 0$; consequently, the tuned error signal and the tuned regressor signal are

$$e_* = H(\theta_*)w = H_{ew}\, w \tag{2.8}$$

$$z_* = H_{zw}\, w \tag{2.9}$$

, respectively. The interconnection system can also be written as

$$e = e_* - H_{ev}\, v \quad , \qquad z = z_* - H_{zv}\, v \tag{2.10}$$

with v given by (2.6).

C. Adaptive Algorithm

To compleat the error model, it is necessary to describe the adaptive algorithm, i.e., the means by which $\hat{\theta}(t)$ is generated. We will denote the relation among the signals e, z, $\hat{\theta}(0)$ and $\hat{\theta}(t)$ as follows

$$u = \Omega(e,\ z,\ \hat{\theta}(0)) \tag{2.11}$$

and consider the most typical algorithm (Fig.2).

$$\tfrac{d}{dt}\hat{\theta} = \Gamma Z^T e \ , \quad \Gamma \in R^{k \times k} \ , \Gamma = \Gamma^T > 0 \ . \tag{2.12}$$

Since this algolithm has a infinite gain and causes instability phenomena with the existence of the plant uncertainty, we modify this adaptive algorithm as shown in Fig.3. We can make the feedback signal:

$$\hat{e} = e - Du \tag{2.13}$$

where $D \in R^{m \times m}$ $D + D^T > 0$, because the signals (e, z, u, r) are all measurable, and will use $\hat{e}$ as the input of the algorithm (2.12). Then the equation (2.12) can be described as

$$\tfrac{d}{dt}\hat{\theta} = -\Gamma Z D Z^T \hat{\theta} + \Gamma Z e \tag{2.14}$$

or, by interconnection structure,

$$\tfrac{d}{dt}\theta = -\Gamma Z D Z^T \theta + \Gamma Z(e - D Z^T \theta_*). \tag{2.15}$$

Note that these algorithm (2.14) and (2.15) have nonlinear decay terms $-\Gamma Z D Z^T \hat{\theta}$ and $-\Gamma Z D Z^T \theta$, respectively because of the feedback gain D.

We denote this new modified adaptive controller by

$$v = \tilde{\Omega}(\tilde{e},\ z,\ \theta(0)), \quad \text{or} \quad u = \tilde{\Omega}(e,\ z,\ \hat{\theta}(0)) \tag{2.16}$$

in the same form as (2.11), where $\tilde{e} = e - D Z^T \theta_*$.

The adaptive error system which uses the adaptive controller (2.16) is shown in Fig.4.

Ⅲ. CONDITION FOR GLOBAL STABILITY

A. Conditons of Global Stability

The adaptive error system (Fig.4) can be transformed to the equivalent feedback system, shown in Fig.5, where operators and signals are defined as follows.

$$H_{zwD}\, w = D\{\mathrm{Mat}(H_{zw}\, w)\}^{T}\ \theta_{*} \tag{3.1a}$$

$$H_{zvD}\, v = D\{\mathrm{Mat}(H_{zv}\, v)\}^{T}\ \theta_{*} \tag{3.1b}$$

$$H_{D} = H_{ev} - H_{zvD} + D = H_{evD} + D \tag{3.1c}$$

$$e_{D*} = (H_{ew} - H_{zwD})\, w\ . \tag{3.2}$$

where Mat(•) denotes the matrix formation corresponding to the matrix Z. Note that the equivalent system is composed of two subsystems: a linear subsystem and a nonlinear subsystem with external inputs.

Because the equivalence of the systems, we can analyze the stability of the adaptive system by the equivalent system (Fig.5). The theorm stated below gives conditions for which the equivalent system is guaran--teed to have certain stability and performance properties. (We use the notaion "$x \to 0(\exp)$" to mean that $x(t) \to 0$(exponentially) as $t\to\infty$.)

Theorem.1 [Global Stability]: For the equivalent system shown in Fig.5 assume the following.

A1: The system is well posed in the sense that all inputs $w\in W$ produce signals e, v, z, θ, and $\dot{\theta}$ in $L_{\infty e}$. (3.3a)

A2: $H_{zv}\in S_{0}^{k\times m}$. (3.3b)

A3: $H_{D}\in SPR_{+}^{m}$. (3.3c)

under these conditions, the following holds.

1) If $(e_{D*}, \dot{e}_{D*})\in L_{2}^{m}\cap L_{\infty}^{m}$ $(= e_{D*} \to 0)$ and $(z_{*}, \dot{z}_{*})\in L_{\infty}^{k}$, then the following holds.

1-a) $(\theta, \ddot{\theta})\in L_{\infty}^{k}$, $\dot{\theta}\in L_{2}^{k}\cap L_{\infty}^{k}$, and $\dot{\theta} \to 0$. (3.4a)

1-b) $\widehat{e}\in L_{2}^{m}\cap L_{\infty}^{m}$, $\widehat{e}\in L_{\infty}^{m}$, and $\widehat{e} - e_{D*} \to 0$. (3.4b)

1-c) $v\in L_{2}^{m}\cap L_{\infty}^{m}$, $\dot{v}\in L_{\infty}^{m}$, and $v \to 0$. (3.4c)

1-d) $(z, \dot{z})\in L_{\infty}^{k}$, $(z-z_{*}, \dot{z}-\dot{z}_{*})\in L_{2}^{k}\cap L_{\infty}^{k}$, and $z-z_{*} \to 0(\exp)$. (3.4d)

1-e) If, in addition, $e_{D*} = 0$ and $z_{*}\in PE$ (persistently exciting) then $(\theta, \dot{\theta}, \widehat{e}-e_{D*}, v, z-z_{*}) \to 0(\exp)$. (3.4e)

2) If $e_{D*}\in L_{\infty}^{m}$, and $z_{*}\in L_{\infty}^{m}$, then the following holds.

2-a) $z\in L_{\infty}^{k}$. (3.5a)

2-b) If, in addition, $z\in PE$, then $(\theta, \dot{\theta})\in L_{\infty}^{k}$, $\widehat{e}\in L_{\infty}^{m}$, and $v\in L_{\infty}^{m}$. (3.5b)

■

(The proofs are entirely analogous to that of in [3] and omitted here.)

B. Robustness to Unmodeled Dynamics

Since the equivalent system shown in Fig.5 entirely represent the properties of the adaptive system, we can analyze the robustness by the invariability of the required conditions on operators and signals in Theorem.1 to unmodeled dynamics in the plant. In particular since the condition $H_D \in SPR_+^m$ is very crucial, we must investigate its robustness the first thing. The invariability of the other conditions, e.g., $H_{zv} \in S_0^{k\times m}$, etc, is not discussed here because these can be treated in the robustness problem of linear systems. Because the plant uncertainty affects operators H_{ev}, H_{zv} ect, suppose that H_{evD} has the form

$$H_{evD} = (I + \tilde{H}_{evD}) \bar{H}_{evD} \tag{3.6}$$

where $\tilde{H}_{evD}$ is the projection onto H_{evD} of the plant uncertainty operator $\Delta \in S^{n\times n}$:

$$P = (I + \Delta) P_{*} \quad , \quad \Delta \in B_S(r) \tag{3.7}$$

where $P \in R(s)^{n\times m}$ is the plant, $P_{*} \in R(s)^{n\times m}$ is the tuned parametric model, and $B_S(r)$ denote a “ball” in $S^{n\times n}$ of radius r , defined by

$$B_S(r) = \{ G \in S^{n\times n} \mid \bar{\sigma}[G(j\omega)] \leqq r(\omega), \ \forall \omega \in R \} \tag{3.8}$$

: and $\bar{H}_{evD}$ is the nominal transfer function when there is no uncertainty i.e., $\Delta = 0$. ($\bar{\sigma}(A)$ denotes the maximum singular value of the matrix A.) Note that $\bar{H}_{evD}$ is a function of the tuned parametric model P_{*} , the tuned controller gains θ_{*} , and the feedback gain D in the new modified adaptive controller (2.16).

Conditions to ensure that $H_D \in SPR_+^m$ despite uncertainty in P is provided by the following.

Theorem.2 [Robustness]: Let H_{evD} be given by (3.6). Then $H_D \in SPR_+^m$ if the following conditions hold :

1) $\bar{H}_{evD} \in SPR_0^m$. (3.9a)

2) $\tilde{H}_{evD} \in B_S(k)$ for $\forall \omega \in R$ where (3.9b)

$$k(\omega) < 1/2 \ \underline{\lambda}[\bar{H}_D(j\omega) + \bar{H}_D(-j\omega)^T] \ / \ \bar{\sigma}[\bar{H}_{evD}(j\omega)] \tag{3.9c}$$

and

$$\bar{H}_D = \bar{H}_{evD} + D \ . \tag{3.9d}$$

■

Proof : Define $\mu(\cdot): C^{m\times m} \rightarrow R$ by

$$\mu(A) = 1/2 \ \underline{\lambda}(A + A^{*})$$

where $*$ denotes complex conjugate transpose. Then using definitions, we obtain

$$\mu[H_D(j\omega)] = \mu[\bar{H}_{evD}(j\omega) + \tilde{H}_{evD}(j\omega) \bar{H}_{evD}(j\omega) + D]$$

$$\geqq \mu[\bar{H}_{evD}(j\omega) + D] - \bar{\sigma}[\tilde{H}_{evD}(j\omega)] \ \bar{\sigma}[\bar{H}_{evD}(j\omega)] > 0 \ .$$

Hence, $H_D \in SPR_+^m$. ■

In the scalar case, (3.9c) becomes

$$k(\omega) < \mathrm{Re}[\ \bar{H}_D(j\omega)\ ] \,/\, |\ \bar{H}_{evD}(j\omega)\ | \qquad (3.10)$$

Since $\bar{H}_{evD} \in SPR_0$ by assumption and $D > 0$, $k(\omega)$ is always positive for $\forall\omega \in R$ and when $\omega \to \pm\infty$, then $k(\omega) \to +\infty$. So we can find that the "ball" $B_s(k)$ contains the wide class of uncertain operators.

C. Discussion

1) From Theorem.2 , if unmodeled dynamics is included in the class which is described by eqs. (3.9b) and (3.9c), $H_D \in SPR_+^m$ holds with the condition $\bar{H}_{evD} \in SPR_0^m$. Thus, from Theorem.1, the signals ($\hat{e}$, v, z) and the adaptive gain error θ of the adaptive error system (Fig.4) are globally bounded with the other restrictions on the tuned signals e_{D*} and z_* , which are indirectly imposed on H_{ew}, H_{zw}, and H_{zwD}. Because the tuned gain θ_* is selected to be bounded, then adaptive gain vector $\hat{\theta}$ is globally bounded.

2) In practice, the case of Theorem 1 - 2) is more realistic because the plant uncertainty also affects operators H_{ew} and H_{zwD}, and e_{D*} does not go to 0. In this case, the weak assumption $z \in PE$ are required to insure that $(\theta, \dot{\theta}) \in L_\infty^k$, $\hat{e} \in L_\infty^m$, and $v \in L_\infty^m$. However since z is inside the adaptve loop, it is not possible to guarantee $z \in PE$ without further ristrictions on the initial parameter error $\theta(0)$ or the spectrum of the external signals. Instead of $z \in PE$, the use of the algorithm with retar-dation also gives the bounded property of $(\theta,\ \dot{\theta},\ \hat{e},\ v)$ ([3]).

3) D is the design parameter to insure the robustness of adaptive system (i.e., D must be selected so that the conditon of eqs. (3.9b) and (3.9c) hold for the unmodeled dynamics in P.), and in general, some information of θ_* is required for its selection. In this meaning, Theorem.1 and Theorem.2 do not give the global robust stability con-dition for $\hat{\theta}$.

From the relation of eq. (2.13) the error signal e(t) does not go to 0 whereas $\hat{e}(t) \to 0$ $(t \to 0)$ in the case of Theorem. 1 - 1). Therefore we can find that the constant error remains on e(t).

These problems are discussed later again.

IV. EXAMPLES

A. Example of Design

We consider the model reference adaptive control (MRAC) system de-picted in Fig.6 consisting of : a scalar plant $P \in R_0(s)$ with unmodeled dynamics; a reference model $H_r \in S_0$; and an adaptive gain $\hat{\theta}(t)$ as a feedback to the plant. The plant is affected by a disturbance d and a reference command r. For example, assume that P and H_r are described as follows.

$$H_r(s) = \frac{1}{s + b} \qquad b>0 \qquad (4.1a)$$

$$P(s) = \frac{1}{\mu s + 1}\ \frac{1}{s + a} \qquad (4.1b)$$

where μ is a small positive parameter.

And assume to use the adaptive algorithm given by (2.14). Thus

$$\tfrac{d}{dt}\hat{\theta} = -\Gamma D y^2 \hat{\theta} + \Gamma y e \ , \qquad \Gamma > 0,\ D > 0 \ . \tag{4.2}$$

Since μ is small, thus we can assume that the tuned parametric model P_* is described as follows.

$$P_*(s) = \frac{1}{s + a} \ . \tag{4.3}$$

For the simplicity we assume that $d = 0$ (i.e.,disturbance free case). Let r = constant ($\neq 0$) and $\Delta = 0$ (i.e., $\mu = 0$). Thus $e_{*D} \rightarrow 0$ (exp) (i.e., $(e_{D*}, \dot{e}_{D*}) \in L_2^m \cap L_\infty^m$) when the tuned gain is

$$\theta_* = (b - a) \ / \ (1 + D b) \tag{4.4}$$

From this tuned gain,

$$H_{ev}(s) = \frac{1}{\mu s^2 + (\mu a + 1)s + b(1 + Da)/(1 + Db)} \tag{4.5a}$$

and

$$H_{evD}(s) = [1 - \frac{b - a}{1 + Db}] H_{ev}(s) \tag{4.5b}$$

holds.

Assume that the value of $(b - a)$ is not so large and $1 + Da > 0$. Then $\bar{H}_{evD} \in SPR_0$ holds because $H_{ev}(s)|_{\mu=0} \in SPR_0$. Denote

$$\inf_{\omega} \mathrm{Re}\, H_{ev}(j\omega) = \nu < 0 \tag{4.5c}$$

and if $\nu > -1$, then $H_{ev}(s) = H_{zv}(s) \in S_0$ holds. Thus some sufficient conditions for the design parameter D to hold that $H_D \in SPR_+$ are as follows.

1) If $a < 0$ i.e., the plant is unstable, then

$$-\nu < D < -1 \ / \ a \ . \tag{4.6a}$$

2) If $a \geqq 0$, and $b - a > 0$ i.e., the time-constant of the model is larger than that of the plant, then

$$-\ \nu\ < D \ . \tag{4.6b}$$

B. Numerical Example

As a numerical example, we take that $a = 1$; $b = 1.5$; and $\mu = 0.2$, (in this case $\nu \fallingdotseq -0.07$: $D = 0$) and simulate this MRAC system on the analog computer. Simulations run with $r = 0.5 \sin 3.8t$, and next three cases are shown in Fig.7, 8, and 9 .

Case 1) With no uncertainty (i.e., $\mu = 0$) (Fig.7).

Case 2) Wiht uncretainty (i.e., $\mu = 0.2$) and $D = 0$ (Fig.8).

Case 3) Wiht uncretainty and $D = 0.15$ (Fig.9).

In Fig.7, we see that $\hat{\theta}(t)$ is stable and the error signal $e(t)$ is very small, i.e., $y(t)$tracks $y_r(t)$ very well. But in Fig.8, $\hat{\theta}(t)$ is unstable and large oscillations are taking place in $e(t)$. Adding $D = 0.15$ which satisfies the condition (4.6b) to case 2), Fig.9 shows that $\hat{\theta}(t)$ is stable while large osillations still exist in e(t).

C. Discussion

1) In this numerical example, the value of tuned gain $\theta_* \fallingdotseq 0.41$ with $r = \text{constant}(\neq 0)$ from eq.(4.4). And when $\hat{e} \to 0$, and $\hat{\theta} \to \theta_*$ $(t \to \infty)$

$$e(t) \to D\, z_*(\infty)\theta_* \fallingdotseq 0.06\, y_*(\infty)\ , \quad (t \to \infty) \tag{4.7}$$

then the constant error on e(t) in this case is about 6 % of the final value of the plant output .

2) It is clear also from eq.(4.4) that the tuned gain θ_* becomes a function of the design parameter D in general whenever we use the new modified adaptive algolithm (2.14). And to select D which satisfies the condition (4.6a) or (4.6b), we need to know the approximate estimate of θ_* (the approximate value of $|a|$ in this example).

3) The conditions of eqs. (4.6a) or (4.6b) in this example are very suggestive. In the first instance, when the parameter μ is large (i.e., the degree of the plant uncertainty is large), we must select D large to get robust properties of the adaptive system with the drawback of the large constant error on e(t). When $a < 0$ and its absolute value is large (i.e., the difference between the model of eq.(4.1a) and the tuned parametric model P_* of eq.(4.3) is large), the maximum value of the allowable D becomes small. This means that the adaptation of far differ--ent systems becomes inherently difficult by the plant uncertainty.

V. CONCLUSIONS

1) In this paper, we proposed a new modified adaptive algorithm (2.14) which was compensated by feedback gain D and proved the MRAC system which used this algorithm had the robust global stability properties to the plant uncertainty. The drawbacks when we use this algorithm, are that some information for the tuned gain θ_* is required in selecting the design parameter D and that the constant error remains on the error signal e(t).

2) It is shown in [6] that the adaptive controller (2.11) which uses the algorithm (2.12) and also the algorithm itself have infinite gains. Ioannou and Kokotovic use the modified adaptive algorithm described by eq.(5.1) in their analysis [1],[2].

$$\frac{d}{dt}\hat{\theta} = -\sigma\hat{\theta} + \Gamma Ze \tag{5.1}$$

This algorithm has a linear decay term $-\sigma\hat{\theta}$ and by virtue of this, the infinite gain of eq.(2.12) is diminished to a finite gain and the MRAC system becomes to have a attractive region. But because the gain of adaptive controller (2.11) which uses the algorithm (5.1) is not bounded their result of stability is local.

3) The new modified adaptive algorithm (2.14) proposed in this paper also have its nature in the diminishing of the infinite gain of eq. (2.15). The difference of modification between eqs.(2.14) and (5.1) is that the decay term in eq.(2.14) is nonlinear with regressor signal and design parameter D , while linear in eq.(5.1).

References

[1] P.A.Ioannou and P.V.Kokotovic, "Instabilit Analysis and Improvement of Robustness of Adaptive Control," Automatica, Vol.20, No.5, pp. 583-594, 1984.

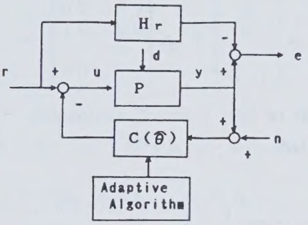
[2] P.A.Ioannou and P.V.Kokotovic, Adaptive Sytems with Reduced Models (Springer-Verlag Series, Lecture Notes inControl and Information Siences, Vol.47), New york, Springer-Verlag, 1983.

[3] R.L.Kosut and B. Friedlander, "Robust Adaptive Control: Conditons for Global Stability," IEEE Trans. Automat. Contr., Vol. AC-30, No.7, July 1985.

[4] C.A.Desoer and M.Vidyasagar, Feedback System: Input-Output Properties. New York: Academic, 1975.

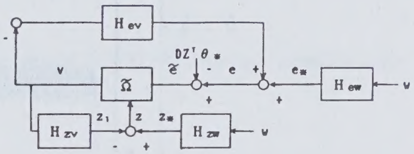
[5] K.S.Narendra and L.S.Valavani, "Direct and Indirect Model Reference Adaptive Control," Automatica, Vol.15, pp.653-664, 1979.

[6] C.E.Rohrs, L.Valavani, M.Athans, and G.Stein "Robustness of Con--tinuous-Time Adaptive Control Algorithms in the Presence of Unmodeled Dynamics," IEEE Trans. Automat. Contr., Vol. AC-30, No.9, Sept. 1985.



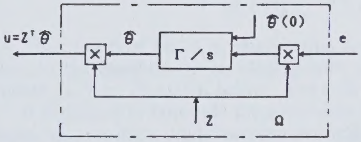
H r
r
+
u
d
P
y
+
-
e
-
C(θ̂)
+
n
+
Adaptive Algorithm

Fig.1



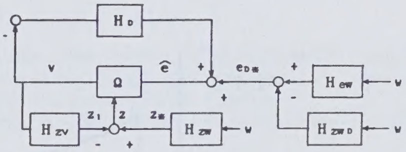
-
H ev
v
DZᵀθ*
ẽ
-
e
+
e*
H ew
w
Ω̃
+
+
z₁
z
z*
H zv
H zw
w
-
+

Fig.4



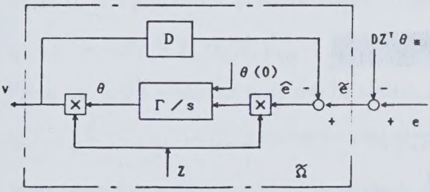
u=Zᵀθ̂
θ̂
θ̂(0)
×
Γ/s
×
e
Z
Ω

Fig.2



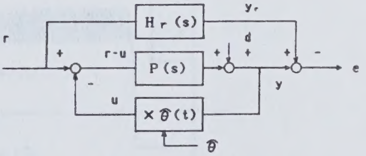
-
H D
v
Ω
ê
+
e D*
+
H ew
w
+
-
z₁
z
z*
H zv
H zw
w
H zwD
w
-
+

Fig.5



D
DZᵀθ*
v
θ
θ(0)
×
Γ/s
×
ê
-
ẽ
-
+
+
e
Z
Ω̃

Fig.3



y r
H r (s)
r
+
r-u
d
P(s)
+
+
+
-
e
-
y
u
× θ̂(t)
θ̂

Fig.6

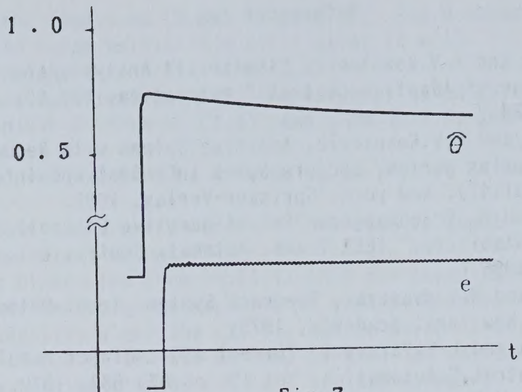


Fig.7

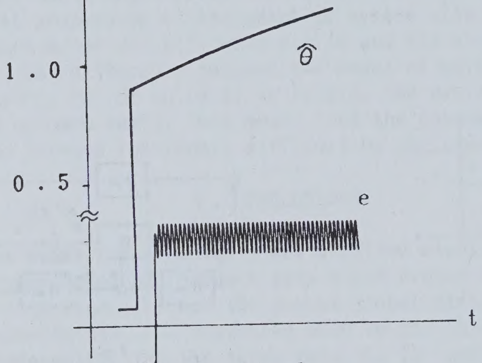


Fig.8

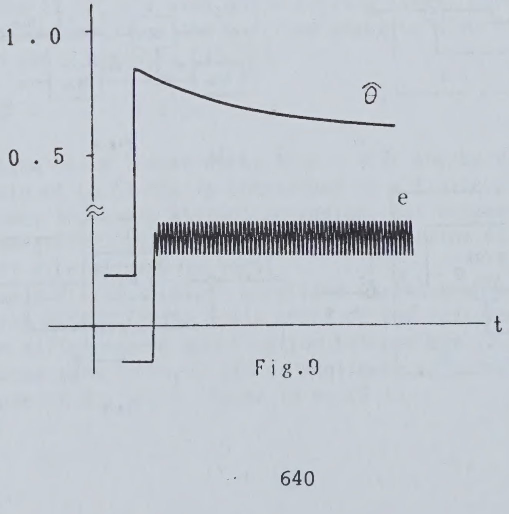


Fig.9

DEADBEAT PARAMETER IDENTIFICATION OF DARMA PROCESSES

H. F. VANLANDINGHAM and MARK A. HOPKINS
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061

This paper presents a deadbeat method of joint parameter and state identification, based on a canonic form given by Salut[1], for observable deterministic invariant discrete-time SISO ARMA systems. Such systems can be represented by the transfer function

$$H(z) = \frac{Y(z)}{U(z)} = \frac{b_n + b_{n-1} z^{-1} + \cdots + b_1 z^{-n+1} + b_0 z^{-n}}{1 - a_{n-1} z^{-1} - \cdots - a_1 z^{-n+1} - a_0 z^{-n}} \tag{1}$$

where the a_i and b_i are system parameters. The observable canonic state model is

$$\begin{aligned} x_{k+1} &= A x_k + B u_k \\ y_k &= C x_k + D u_k \end{aligned}$$

$$\textit{where}\quad A = \begin{bmatrix} 0\,0 \ldots 0\,0\; a_0 \\ 1\,0 \ldots 0\,0\; a_1 \\ \vdots \\ 0\,0 \ldots 0\,1\; a_{n-1} \end{bmatrix}, \qquad B = \begin{bmatrix} b_0 + a_0 b_n \\ b_1 + a_1 b_n \\ \vdots \\ b_{n-1} + a_{n-1} b_n \end{bmatrix}, \tag{2}$$

$$C = [\,0\,0 \ldots 0\,0 \quad 1\,], \quad \textit{and}\ D = [\,b_n\,]$$

The first line of Eq. 2 can be rewritten as

$$x_{k+1} = \begin{bmatrix} 0\,0 \ldots 0\,0 \\ 1\,0 \ldots 0\,0 \\ \vdots \\ 0\,0 \ldots 1\,0 \end{bmatrix} x_k + \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{bmatrix} y_k + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{bmatrix} u_k \overset{\Delta}{=} J x_k + p_a y_k + p_b u_k \tag{3}$$

Equations 2 and 3 lead to the extended state model (ESM). Define the extended state, $s_k \overset{\Delta}{=} [\, x_k^T \mid p_a^T \mid p_b^T \mid b_n \,]^T$; note that $s_k \in R^{3n+1}$. Thus,

$$\begin{aligned} s_{k+1} &= F(y_k, u_k)\, s_k \overset{\Delta}{=} F_k s_k \\ y_{k+1} &= H(u_{k+1})\, s_{k+1} \overset{\Delta}{=} H_{k+1} s_{k+1} \\ &\textit{where}\quad F_k = \left[\begin{array}{c|c|c|c} J & y_k I_n & u_k I_n & 0 \\ \hline 0 & \multicolumn{3}{c}{I_{2n+1}} \end{array}\right] \\ &\textit{and}\ H_{k+1} = [\,0 \ldots 0\,1 \mid 0 \ldots 0\,0 \mid 0 \ldots 0\,0 \mid u_{k+1}\,] \end{aligned} \tag{4}$$

The system of Eq. 4 is time-varying and autonomous. The question of its observability is equivalent to the question of the identifiability of the system of Eq. 1, so is clearly related to the input sequence $\{u_k\}$, as discussed below. For now, assume $\{u_k\}$ is a suitable input sequence.

It is desired to compute the minimum mean square error (MMSE) recursive prediction of the state vector s_{k+1}, using a predictor/corrector algorithm,

$$\hat{s}_{k+1|k} = F_k \hat{s}_{k|k}, \quad \textit{and} \quad \hat{s}_{k|k} = \hat{s}_{k|k-1} + \bar{G}_k (y_k - H_k \hat{s}_{k|k-1}) \tag{5}$$

where $\bar{G}_k$ is a gain vector to be computed. Equation 5 can be written as

$$\hat{s}_{k+1|k} = (F_k - F_k \bar{G}_k H_k)\hat{s}_{k|k-1} + F_k \bar{G}_k y_k \overset{\Delta}{=} (F_k - G_k H_k)\hat{s}_{k|k-1} + G_k y_k \tag{6}$$

From Eq. 6, the prediction error can be written recursively as

$$e_{k+1|k} \overset{\Delta}{=} \hat{s}_{k+1|k} - s_{k+1} = (F_k - G_k H_k) e_{k|k-1} \tag{7}$$

Now define the prediction error outer product, $P_{k+1|k} \overset{\Delta}{=} e_{k+1|k}\, e_{k+1|k}^T$; so, from Eq. 7,

$$P_{k+1|k} = F_k P_{k|k-1} F_k^T - G_k H_k P_{k|k-1} F_k^T - F_k P_{k|k-1} H_k^T G_k^T + G_k H_k P_{k|k-1} H_k^T G_k^T \tag{8}$$

Minimization with respect to G_k is performed on the functional $L_{k+1|k} \overset{\Delta}{=} \|e_{k+1|k}\|^2 = \text{trace}(P_{k+1|k})$.

Denoting the minimizing value of the gain G_k by $K_{k+1|k}$, the result is

$$K_{k+1|k} = F_k P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T)^{\#} \qquad (9)$$

where "#" indicates the pseudo-inverse. Since the quantity to be inverted is scalar, its pseudo-inverse is the reciprocal, if it exists, and is zero otherwise. Substituting Eq. 9 into Eq. 8,

$$P_{k+1|k} = (F_k - K_{k+1|k} H_k) P_{k|k-1} F_k^T , \quad \text{where } P_{0|-1} \text{ is symmetric positive definite.} \qquad (10)$$

Proofs of the following lemmas are available upon request:

Lemma 1 If $P_{0|-1}$ is positive definite symmetric, and $\text{rank}(P_{k|k-1}) = m > 0$, then
a) If $H_k P_{k|k-1} H_k^T \neq 0$, then $\text{rank}(P_{k+1|k}) = m - 1$,
and b) If $H_k P_{k|k-1} H_k^T = 0$, then $\text{rank}(P_{k+1|k}) = m$.
Also, $(P_{k|k-1} = 0) \rightarrow (P_{k+1|k} = 0)$.

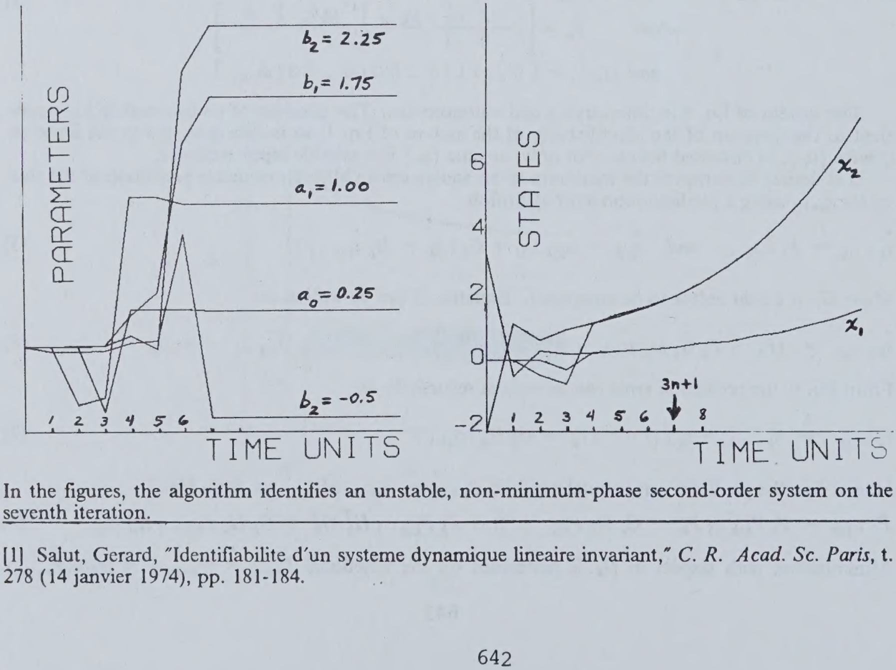
Lemma 2 If $P_{0|-1}$ is positive definite symmetric, and $P_{k|k-1} = 0$, then $e_{k|k-1} = 0$, for arbitrary $e_{0|-1}$.

From lemmas 1 and 2, it is clear that the convergence of the predictor depends upon the input sequence. However, the algorithm will converge no matter how long it takes to obtain a sufficiently exciting input. Furthermore, it is simple to determine when convergence has occurred, viz., by counting the number of times that $H_k P_{k|k-1} H_k^T$ is non-zero. When the count reaches $3n + 1$, convergence has occurred. Each delay of convergence is analogous to requiring another row for a generalized inverse in an off-line least squares algorithm.

Elsewhere, the authors have demonstrated equivalence between the non-zero value of $H_k P_{k|k-1} H_k^T$ and the linear independence of the associated (bottom) row of the observation matrix Θ_k, where

$$\xi^k \overset{\Delta}{=} \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} H_0 \\ H_1 F_0 \\ \vdots \\ H_k F_{k-1} \cdots F_0 \end{bmatrix} s_0 \overset{\Delta}{=} \Theta_k s_0 \qquad (11)$$

Thus, as soon as Θ_k attains rank of $3n + 1$, i.e., as soon as its generalized inverse exists, the algorithm described by Eqs. 6, 9, and 10 converges. Thus it is deadbeat, since $s_k = F_{k-1} \cdots F_0 (\Theta_k^T \Theta_k)^{-1} \Theta_k^T \xi^k$ is not computable before the inverse exists.



In the figures, the algorithm identifies an unstable, non-minimum-phase second-order system on the seventh iteration.

---

[1] Salut, Gerard, "Identifiabilite d'un systeme dynamique lineaire invariant," *C. R. Acad. Sc. Paris*, t. 278 (14 janvier 1974), pp. 181-184.

ADAPTIVE CONTROL OF A CLASS OF UNCERTAIN DYNAMICAL SYSTEMS

F. POURBOGHRAT AND A. DATTA

Department of Electrical Engineering
Southern Illinois University
Carbondale, IL 62901

ABSTRACT

A class of uncertain dynamical systems is considered. An adaptive controller is designed which guarantees the stability of the system under bounded uncertainties. The rate of convergence for the adaptation law may be preselected. The proposed controller does not require the knowledge of system parameters, and can easily be implemented.

SUMMARY

Control of uncertain nonlinear systems is an important subject of research from both practical and scientific points of view. Some of the results which are now available are based on the min-max differential game theory [1-3]. In this paper, a class of uncertain dynamical systems is considered. A simple adaptive controller is designed, [3], which does not require the knowledge of system parameters, and guarantees the stability of the system. It is also possible to preselect the rate of convergence for the adaptation process.

We consider the dynamical system of the type

$$\dot{x} = f(t,x) + Bg(t,x,u) \tag{1}$$

where $x \in R^n$ is the state vector, $u \in R^m$ is the input vector, $f\colon RxR^n \to R^n$, $g\colon RxR^n xR^m \to R^m$, and $B = [0,I]^T$. Moreover, we assume that f is Caratheodory, [2], and there is a vector $q(t,x)$ such that

$$f(t,x) + Bq(t,x) = Ax$$

where $q\colon RxR^n \to R^m$ is strongly Caratheodory, [2], and A is a stable matrix, with desired eigenvalues. Also, g is strongly Caratheodory and there is a strongly Caratheodory positive function $h(t,x)$ and an unknown constant $b_0 > 0$ such that for all $(t,x,u) \in RxR^n xR^m$, $u^T g(t,x,u) \geq b_0 ||u||[||u|| - h(t,x)]$.

PROPOSED CONTROLLER

For the above class of uncertain nonlinear dynamical systems, we propose the following controller:

$$u(t) = \begin{cases} -k(t)\, a/||a|| & , \quad a \neq 0 \\ 0 & , \quad a = 0 \end{cases} \tag{2}$$

where $a = B^T Px$ such that $PA + A^T P = -Q$ and P,Q are positive definite matrices, and $k(t) = k_0 + k_1 V + k_2 \int_0^t V$ with $k_0, k_1, k_2 > 0$ and $V = x^T Px$.

Result: Consider the uncertain nonlinear dynamical system (1), driven by the controller (2). Then the controlled system has a solution and it is globally asymptotically stable about the zero equilibrium point. Moreover, the rate of convergence of the adaptation process may be adjusted by properly selecting the matrix A with desired eigenvalues.

Proof: The existence of solution can be shown by following the arguments in [2]. To show the global asymptotic stability of the system, we first note that the system (1) can be written as

$$\dot{x} = Ax + Bg_1(t,x,u)$$

where A is stable and has desired eigenvalues, and g_1 is strongly Caratheodory such that $u^T g_1(t,x,u) \geq b_0||u||[||u||-h_1(t,x)]$ with $h_1(t,x)$ bounded. Now, define the Liapunov function $V = x^T Px$, such that $PA + A^T P = -Q$ where P,Q are positive definite matrices. Then, with the controller (2), we have

$$\begin{aligned}\dot{V} &= -x^T Qx + 2x^T PBg_1(t,x,u)\\ &\leq -x^T Qx + 2a^T g_1 \leq -x^T Qx - 2b_0||a||[||u||-h_1].\end{aligned}$$

In the above, the last term is negative if $k(t) = ||u|| > h_1$. However, since V is positive, there is a finite time t_1 such that $k(t_1) > h_1$.

Therefore, $\dot{V} < 0$ and the global asymptotic stability of the zero equilibrium point is guaranteed. If $k_0 \geq h_1$, it is also guaranteed that the rate of convergence of the adaptation process is according to the prefixed eigenvalue $\lambda = \lambda_{min}(Q)/\lambda_{max}(P)$, which is dependent on the eigenvalues of the preselected matrix A, [3].

It is to be mentioned that given system (1), in practice, for a preselected matrix A with desired eigenvalues, we end up with a vector q(t,x) which is not strongly Caratheodory. In fact, often q(t,x) and hence $h_1(t,x)$ are only Caratheodory functions. Although this does not alter the stability properties of the system, it does put a constraint on achieving a desired rate of convergence for the adaptation process.

REFERENCES

[1] Gutman, S., "Uncertain Dynamical Systems - A Lyapunov Min-Max Approach", IEEE Trans. Auto. Control, Vol. AC-24, No. 3, 1979.

[2] Corless, M. and Leitmann, G., "Adaptive Control of Systems Containing Uncertain Functions and Unknown Functions with Uncertain Bounds", J. of Optimization Theory and Applications, Vol. 41, No. 1, 1983.

[3] Pourboghrat, F., "An Adaptive Nonlinear Control for Robotic Systems with Uncertain Parameters", 29th Midwest Symp. on Circuits and Systems, Proceedings, 1986.

ADAPTIVE CONTROL OF DISCRETE TIME-INVARIANT PROCESSES HAVING VARIABLE TIME DELAY

HEUNG-SUK YANG, YOUNG-CHOL KIM AND KOOK-HUN KIM
Dept. of Electrical Eng., Seoul National Univ., Seoul, Korea

ABSTRACTS

A new on-line estimation algorithm for variable time delay is proposed. The basic idea of this algorithm is that the prediction error corresponding to the true dead time is the minimum. Since an indirect predictor is employed, the additional computation burdens are negligible even if the system has a large dead time. It is verified that this algorithm estimates the variable time delay accurately. Simulation results show that adaptive controllers using time delay estimator have good performances.

1. INTRODUCTION

Since necessary conditions assumed in adaptive controllers are not generally met in real systems, robustness of adaptive control has become an important topic in adaptive control theory. In standard type self tuning regulators (STR)(1) and self tuning controllers (STC)(2), the time delay of the process is assumed to be known and constant. However, in many cases the time delay is often unknown and/or time varying. It is well known that a mismatch of time delay between process and model may result in poor or unstable performances.(3,4)

In order to handle the variable time delay, two approaches have been developed; one is to estimate the time delay explicitly(3,5), and the other is to extend the zero polynomial to the maximum time delay so that the time delay can be included implicitly in the numerator polynomial of the transfer function.(4,6,7) Such methods have a major drawback that the number of the parameters to be estimated increases with the maximum time delay. Consequently, when the length of time delay is large, the previous methods are less effective for two reasons; 1) the persistently exciting condition in input signals and computational burdens are severe, 2) the adaptive capability of the controller is aggravated since it takes a long time to retune.

In this paper, we propose a new algorithm for the time delay estimation in linear discrete time invriant systems. The basic idea of the algorithm is that the time delay corresponding to the minimum of the prediction error variances is the true time delay of the process.

Main advantage of this algorithm is that the number of parameters to be estimated is independent of the length of the time delay. Simulation results show that the STC and pole assignment self tuning controller (PASTC) using this algorithm improve the performance compared with standard STC and PASTC when the time delay of the process is large and/or varying.

2. TIME DELAY ESTIMATION(TDE)

2.1 Problem Statements

The process to be considered is a SISO linear discrete time system:

$$A(q^{-1})y(t) = q^{-d^0} B(q^{-1})u(t) + w(t) \tag{2.1}$$

where
$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots\ldots\ldots\ldots + a_n q^{-n} \tag{2.2}$$

$$B(q^{-1}) = b_0 + b_1 q^{-1} + \ldots\ldots\ldots\ldots + b_m q^{-m}, \tag{2.3}$$

q^{-1} is a backward shift operator, and d^0 is the time delay of the process assumed to be unknown time varying. y(t) is an output, u(t) is a control input and w(t) is a white noise process with distribution $N(0, \sigma^2)$.

In this paper we don't consider unmodeled dynamics, so the process model is

$$A(q^{-1})y(t) = q^{-d} B(q^{-1})u(t)+w(t), \tag{2.4}$$

where "d" denotes the time delay of the model which has the meaning of the delay variable to be estimated.

2.2 Algorithm

The following assumptions are necessary to establish the new algorithm.
Assumptions :

1. Initial value of d^0 is known.
2. Control input u(t) satisfies the persistently exciting condition, and signal to noise ratio (SNR) is sufficiently large.
3. d^0 is positive integer and changes within the range of $d^0 \pm s$ at a time.

Assumption 1 can be satisfied by some off-line identification methods, and assumption 2 is commonly used in adaptive control. The fractional delay is not considered. "s" indicates the range of delay variation during one sampling interval. Under above assumptions, once the model parameters are estimated by a recursive least square estimator, then (2s+1) indirect adaptive predictors are generated by N (averaging horizon) step past parameters.

The prediction errors are calculated from the predictors, $y(t|t-d)$ and the measured output, $y(t)$, at t. When a dead time change occurs, it is expected that the variance of the prediction error corresponding to the true dead time is the minimum.

Now, a new algorithm for delay estimation is presented as follows.

Algorithm 1

step 1 : Let $d^0 = \hat{d}^0$, where $\hat{d}^0$ is the initial or estimated value of d^0, and identify the parameter,

where $\theta(t-d^0) = (a_1, \ldots\ldots, a_n, b_0, \ldots\ldots b_m)$

step 2 : Construct an indirect predictor set

$$\hat{y}(t|t-d) = G(q^{-1})y(t-d) + F(q^{-1})\hat{B}(q^{-1})u(t-d)$$

$$1 = F(q^{-1})\hat{A}(q^{-1}) + q^{-d}G(q^{-1}), \text{ for } d=d^0-s, \ldots d^0+s$$

where $s \geq 1$ and $\hat{A}(q^{-1})$, $\hat{B}(q^{-1})$ are polynomials at t-N

step 3 : Construct a prediction error set ; $\eta_d(t) = y(t)-\hat{y}(t|t-d)$

step 4 : Determine new estimate of d^0, $\hat{d}^0$, by the criterion.

$$\hat{d}^0 = \min_d E[\eta_d^2(t)] = \min_d [1/N \cdot \sum_{t-N+1}^{t} \eta_d^2(t)]$$

step 5 : Go to step 1.

Remarks :

Variation of the process time delay makes the estimated model parameter deviate rapidly from the process parameter. To check the variation of the time delay, it is recommended that the system parameter used in prediction be the past one, which has been identified before the change of time delay. As the expectation is replaced by a time average, it is necessary to choose the averaging horizon(N) properly. By our experience, $N=n+m+d^0$ is a good one.

3. ANALYSIS OF THE ALGORITHM

In this section, we verify the validity of Algorithm 1. Consider a system described by (2.1) to (2.4). Optimal indirect predictor of the process(8) is

$$\hat{y}(t|t-d^0) = \bar{G}(q^{-1})y(t-d^0) + \bar{F}(q^{-1})\hat{B}(q^{-1})u(t-d^0) \qquad (3.1)$$

$$1 = \bar{F}(q^{-1})\hat{A}(q^{-1}) + q-d^0\,\bar{G}(q^{-1}), \qquad (3.2)$$

where

$$\bar{F}(q^{-1}) = 1 + \bar{f}_1 q^{-1} + \ldots\ldots + \bar{f}_{d^0-1} q^{-d^0+1} \qquad (3.3)$$

$$\bar{G}(q^{-1}) = \bar{g}_0 + \bar{g}_1 q^{-1} + \ldots\ldots + \bar{g}_{n-1} q^{-n+1}. \qquad (3.4)$$

Optimal predictor of the model is

$$\hat{y}(t \mid t-d) = G(q^{-1})y(t-d) + F(q^{-1})\hat{\bar{B}}(q^{-1})U(t-d) \tag{3.5}$$

$$1 = F(q^{-1})\hat{\bar{A}}(q^{-1}) + q^{-d}G(q^{-1}) \tag{3.6}$$

where $$F(q^{-1}) = 1 + f_1 q^{-1} + \ldots\ldots\ldots\ f_{d-1} q^{-d+1} \tag{3.7}$$

$$G(q^{-1}) = g_0 + g_1 q^{-1} + \ldots\ldots\ldots\ g_{n-1}\ q^{-n+1}\ . \tag{3.8}$$

Prediction error $\eta_d(t)$ for an arbitrary time delay d is

$$\eta_d(t) \triangleq y(t) - \hat{y}(t \mid t-d) \tag{3.9}$$

$\hat{y}(t \mid t-d^0)$ is optimal predictor and $\{\hat{y}(t \mid t-d)\}$ is a predictor set constructed to check a delay change.

The optimal prediction error is

$$\begin{aligned} e(t) &= y(t) - \hat{y}(t \mid t-d^0) \\ &= w(t) + \bar{f}_1 w(t-1) + \ldots\ldots + \bar{f}_{d^0-1} \cdot w(t-d^0+1) \end{aligned} \tag{3.10}$$

Substituting (3.10) into (3.9),

$$\eta_d(t) = e(t) + \hat{y}(t \mid t-d^0) - \hat{y}(t \mid t-d) \tag{3.11}$$

From (3.1), (3.5) and (3.11)

$$\begin{aligned} \eta_d(t) = e(t) &+ [q^{-d^0}\bar{G}(q^{-1}) - q^{-d}G(q^{-1})]y(t) \\ &+ [q^{-d^0}\bar{F}(q^{-1})\hat{\bar{B}}(q^{-1}) - q^{-d}F(q^{-1})\hat{\bar{B}}(q^{-1})]u(t) \end{aligned} \tag{3.12}$$

Since control input u(t) can be thought as deterministic, y(t) can be decomposed as follows :

$$y(t) = y_{det}(t) + y_{sto}(t) \tag{3.13}$$

$$y_{det}(t) = [q^{-d^0} B(q^{-1})/A(q^{-1})]u(t) \tag{3.14}$$

$$y_{sto}(t) = [1/A(q^{-1})]\ w(t), \tag{3.15}$$

where $y_{det}(t)$ is deterministic, while $y_{sto}(t)$ is purely stochastic and autoregressive form. $y_{sto}(t)$ can be approximated in moving average form (3.16).

$$y_{sto}(t) = w(t) + \alpha_1 w(t-1) + \alpha_2 w(t-2) + \ldots\ldots \tag{3.16}$$

Substituting (3.13) into (3.12),

$$\eta_d(t) = e_1(t) + e_2(t) + e_3(t), \tag{3.17}$$

where $$e_1(t) \triangleq [g^{-d^0} G(q^{-1}) - q^{-d} G(q^{-1})]\ y_{sto}(t) \tag{3.18}$$

$$e_2(t) \triangleq e(t) \tag{3.19}$$

$$e_3(t) \triangleq [q^{-d^0}\bar{G}(q^{-1}) - q^{-d}G(q^{-1})]\, y_{det}(t) + [q^{-d^0}\bar{F}(q^{-1})\hat{\bar{B}}(q^{-1}) - q^{-d}F(q^{-1})\hat{\bar{B}}(q^{-1})]u(t) \qquad (3.20)$$

and $e_3(t)$ is a deterministic term.

Now we show that algorithm 1 estimates delay accurately. Two cases, $d \geq d^0$ and $d \leq d^0$, are considered in Lemma 1 and Lemma 2.

Lemma 1.

If $d \geq d^0$, $E[\eta_d^2(t)]$ is minimized when $d = d^0$.

proof :

when $d=d^0+1$, (3.3), (3.4), (3.7), (3.8), (3.18) and (3.20) are rewritten as

$$e_1(t) = q^{-d^0}[g_0+(\bar{g}_1-g_0)q^{-1} + \dots\dots+(\bar{g}_{n-1}-g_{n-2})q^{-n+1}+g_{n-1}q^{-n}]y_{sto}(t) \qquad (3.23)$$

$$e_2(t) = e(t) \qquad (3.24)$$

$$e_3(t) = q^{-d^0}[\bar{g}_0+(\bar{g}_1-g_0)q^{-1} + \dots+(\bar{g}_{n-1}-g_{n-2})+g_{n-1}q^{-n}]\, y_{det}(t) \qquad (3.25)$$

If $d = d^0$, because of the facts that $\bar{f}_i$ and $\bar{g}_i$ are calculated from N step past data, $\bar{f}_i = f_i$ and $\bar{g}_i = g_i$. Thus the following is obtained.

$$e_1(t) = q^{-d^0}[\bar{g}_0-g_0)+(\bar{g}_1-g_1)q^{-1} + \dots (\bar{g}_{n-1}-g_{n-1})q^{-n+1}]\, y_{sto}(t)=0 \qquad (3.26)$$

$$e_2(t) = e(t) \qquad (3.27)$$

$$e_3(t) = q^{-d^0}[\bar{g}_0-g_0)+(\bar{g}_1-g_0)q^{-1} + \dots(\bar{g}_{n-1}-g_{n-1})q^{-n+1}]y_{det}(t)=0 \qquad (3.28)$$

In cases of $d \geq d^0$, $e_1(t)$ is composed of $w(t-d^0)$, $w(t-d^0-1)$, ..., and $e_2(t)$ is composed of $w(t)$, ..., $w(t-d^0+1)$. As $e_1(t)$ and $e_2(t)$ are uncorrelated and $e_3(t)$ is deterministic, we get (3.30).

$$E[\eta^2_{d^0+1}(t)] = E[e_1^2(t)] + E[e^2(t)] + e_3^2(t) \qquad (3.30)$$

And $E[e_1^2(t)] + e_3^2(t)] > 0$

$$\therefore\ E[\eta^2_{d^0}(t)] < E[\eta^2_{d^0+1}(t)]. \qquad (3.31)$$

For $d \geq d^0+2$, the same result is easily obtained. Lemma 1 is proved. ***

Lemma 2.

Assuming that SNR is sufficiently large and $d \leq d^0$, $E[\eta_d^2(t)]$ is minimized when $d = d^0$.

proof :

$$E[\eta_d^2(t)] = E[e_1^2(t)] + E[e_2^2(t)] + E[e_3^2(t)] + 2E[e_1(t)e_2(t)] \quad (3.32)$$

$$E[\eta_{d^0}^2(t)] = E[e_2^2(t)] = E[e^2(t)] \quad (3.33)$$

To verify that $E[\eta_d^2(t)] > E[\eta_{d^0}^2(t)]$ for $d<d^0$, the following inequality should be satisfied.

$$E[e_1^2(t)] + E[e_3^2(t)] + 2E[e_1(t)e_2(t)] > 0 \quad \text{for} \quad d \leq d^0-1 \quad (3.34)$$

As only $w(t-d^0+1)$ is a common term in $e_1(t)$ and $e_2(t)$, it is satisfied that

$$-k\sigma^2 \leq 2E[e_1(t)e_2(t)] \leq k\sigma^2 \quad (3.35)$$

where $\quad k = 2 \cdot |\bar{f}_{d^0-1} \cdot g_0|$

In this case, $e_3(t)$ is deterministic,

$$\begin{aligned} e_3(t) = q^{-d^0+1}[-g_0+(\bar{g}_0-g_1)q^{-1} + \ldots(\bar{g}_{n-2}-g_{n-1})q^{-n+1}+\bar{g}_{n-1}q^{-n}] \cdot \\ y_{det}(t) + q^{-d^0}[(1+\bar{f}_1 q^{-1} + ,\ldots \bar{f}_{d^0-1} q^{-d^0+1}) \\ -q(1+f_1 q^{-1}+\ldots.+f_{d^0-2}q^{-d^0+2})]u(t) \end{aligned} \quad (3.36)$$

From (3.34) and (3.35)

$$E[e_1^2(t)] + e_3^2(t) > k\sigma^2 \quad (3.37)$$

As $e_3^2(t)$ in eq. (3.37) implies the power of inputs and outputs, the inequality (3.37) is satisfied for large SNR. ***

Theorem 1 :

Under the previous assumptions, the variance of the prediction error corresponding to the true delay is the minimum. That is

$$d^0 = \min_d E[\eta_d^2(t)] \quad \text{for} \quad d^0-s \leq d \leq d^0+s, \quad \forall t$$

proof :

As results of lemma 1 and lemma 2, it is evident. ***

Algorithm 1 is established by theorem 1.

As only the system parameter (A,B) is to be estimated in algorithm 1, the additional calculation burdens for the prediction error set are trivial. When the delay length is large, the new algorithm is very effective. Moreover the presented method estimates the true delay accurately in the presence of stochastic noise.

4. IMPROVED ADAPTIVE CONTROL

The time delay estimation(TDE) algorithm can be combined with standard adaptive controllers. In this paper, STC and PASTC is considered.

4.1 STC using time delay estimation (STC-TDE)

Using the TDE algorithm for a process with variable time delay, the optimal predictor can be obtained at every step. In generalized minimum variance controller (so called STC), the cost function is given as

$$J = E[1/2(y(t+\hat{d}^0) - y^*(t+\hat{d}^0))^2 + \lambda/2\, u^2(t)] \qquad (4.1)$$

where $y^*(t)$ is the reference output.

From (3.1) and (4.1), the optimal control law is :

$$u(t) = [y^*(t+\hat{d}^0) - G(q^{-1})y(t)] \,/\, [F(q^{-1})\,\hat{B}(q^{-1}) + \Gamma] \qquad (4.2)$$

where $\Gamma = \lambda/\hat{b}_0$.

STC-TDE improves the performance of STC by estimating the variable time delay.

4.2 PASTC using time delay estimation (PASTC-TDE)

The PASTC by wellstead et al.(4) can be applied to the processes that the time delay is unknown and/or time varying. But the order of the controller is increased by the same order. But in PASTC-TDE, the minimum order controller is designed by virture of the on-line estimation of the variable dead time. As a result, computational burden is reduced and the self-tuning capability is still improved when the system dynamics changes.

Consider System (2.1).

With the use of TDE algorithm, the model and the regulator are expressed as

$$\hat{A}(q^{-1})y(t) = q^{-\hat{d}^0}\hat{B}(q^{-1})u(t) + w(t) \qquad (4.3)$$

$$R(q^{-1})u(t) = -S(q^{-1})y(t) + H(q^{-1})y^*(t+\hat{d}^0) \qquad (4.4)$$

The pole assignment equation is obtained from (4.3) and (4.4).

$$\hat{A}(q^{-1})\,R(q^{-1}) + q^{-\hat{d}^0}\hat{B}(q^{-1})\,S(q^{-1}) = T(q^{-1}) \qquad (4.5)$$

where $T(q^{-1})$ is a prespecified polynomials.

From (4.5), we can easily find that the minimum order controller is designed if d^0 is known exactly.

5. SIMULATION RESULTS

Digital computer simulations were performed for the following process.

$$A(q^{-1})y(t) = q^{-d^0} B(q^{-1})u(t) + w(t)$$

where $A(q^{-1}) = 1 - 0.4\, q^{-1} - 0.32\, q^{-2}$

$$B(q^{-1}) = K(0.6 + 0.3\, q^{-1})$$

and K is the process gain, w(t) is white Gaussian noise, and d^0 is sequentially changed 2-1-2-3-4 at every 60 steps.

Fig. 1 is the features of $E[\eta_d^2(t)]$ for d=1,2,3,4. Input signal is $U(t) = [3.1 \sin(3.8t)+\sin(t) - 4.4 \sin(5.3t)+3 \cos(8.4t)]/5 + 1$

Fig. 1 shows the validity of theorem 1. Fig.2 shows that the algorithm 1 (TDE) estimates a variable time delay and parameters well. In Figs. 3-7, d^0 is changed by 2-1-2-3 at every 100 steps. Fig.3 shows the good performance of STC-TDE for a process with variable time delay. Fig.4 and Fig.5 are the features of STC-TDE and standard STC assuming a fixed delay(d^0=2) in case that the gain K is reduced from 1 to 0.6 at the step 160.

In Fig.5, with the standard STC, the system becomes unstable at 120 step when d^0 is 1 and the output is biased under the set point after the gain changes. However STC-TDE shows the good performance as in Fig.4. Figs. 6-7 are the results of PASTC-TDE and conventional PASTC respectively in case that the reference output is constant and the gain K is changed from 1 to 0.6 at the step 150. Fig.6 and Fig.7 show that PASTC needs more strict P.E. condition than PASTC-TDE, and the tracking speed of PASTC-TDE is faster than that of PASTC.

6. CONCLUSION

In this paper, a new on-line estimation algorithm for variable time delay is suggested. Under some assumptions, it is shown that this algorithm (TDE) estimates the time varying time delay accurately.

The computational burdens are reduced compared with other methods, and are independent of the time delay length. Therefore, the larger the time delay is, the more effective TDE is.

Since adaptive controller using TDE has the minimum number of parameters to be adjusted, this makes P.E. condition less restrictive and improves the performance of the controller when the time delay is large or time var varying. Simulation results coincide with the analysis.

7. REFERENCES

1. Astrom K.J. and Wittenmark, B., "On self-tuning regulators". Automatica,

Vol.9, pp.183-189, 1980.

2. Clarke, D.W. and Gawthrop, P.J., "Self-tuning controller", Proc. IEE, Vol.122, No.9, pp.929-934, 1975.

3. Kurz, H. and Goedecke, W., "Digital parameter adaptive control of process with unknown dead time", Automatica, Vol.17, No.1, pp.245-252, 1981.

4. Wellstead, P.E., Prager, D.L. and Zanker, P.M., "Pole assignment self-tuning regulator", Proc. IEE, Vol.128, No.8, pp.781-787, 1979.

5. Wong, K.Y. and Bayoumi, M.M., "A self-tuning control algorithm for systems with unknown time delay", IFAC. Identification and system parameter estimation, Washington D.C., pp.1193-1198, 1982.

6. Ydstie, B.E., "Extended horizon adaptive control", 9th IFAC World Congress, Budapest, Hungary., pp.133-137, 1984.

7. Chien, I.L., Seborg, D.E. and Mellichamp, D.A., "A self-tuning controller for systems with unknown or varying time delays", Int. J.Control, Vol.42, No.4, pp.949-964, 1985.

8. Goodwin, G.C. and Sin, K.S. Adaptive filtering, prediction and control, Prentice-Hall, 1984.

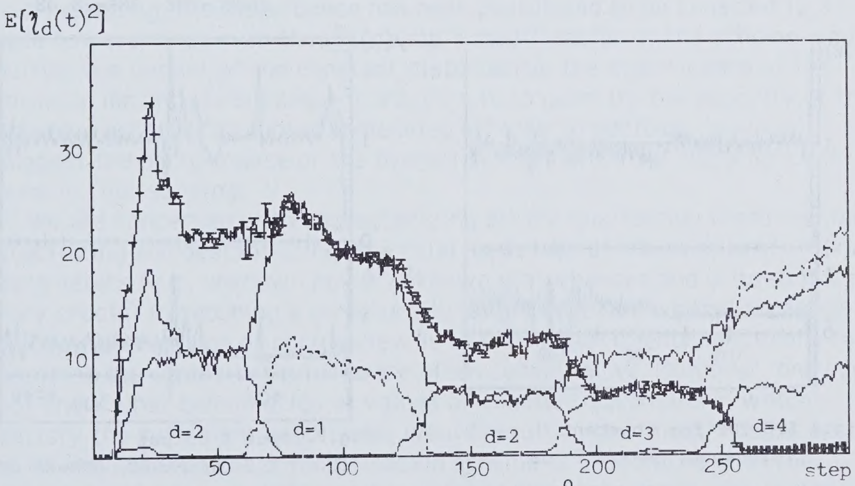


Fig.1 Expectation of Prediction Errors for d_0=1,2,3,4 and SNR= 24.3 dB

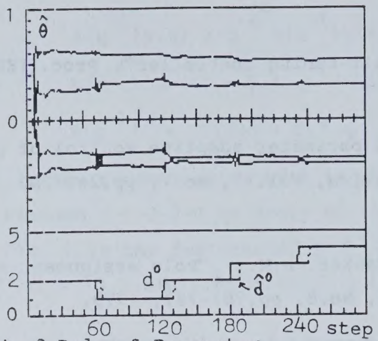


Fig.2 Delay & Parameter
Estimation by TDE Algorithm

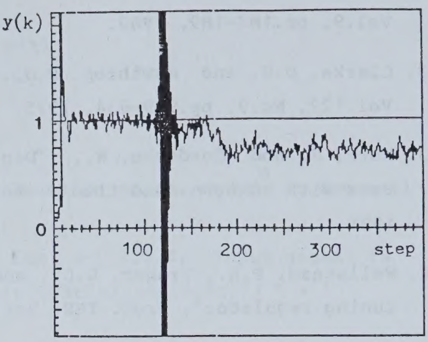


Fig.5 STC without Delay
EStimation, SNR=24.9 dB

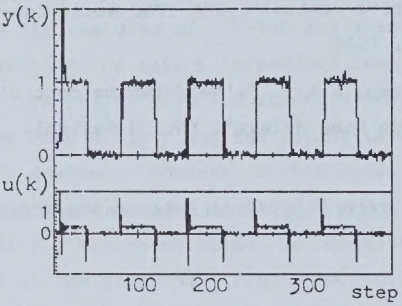


Fig.3 STC-TDE for rectangular
reference, SNR=24 dB

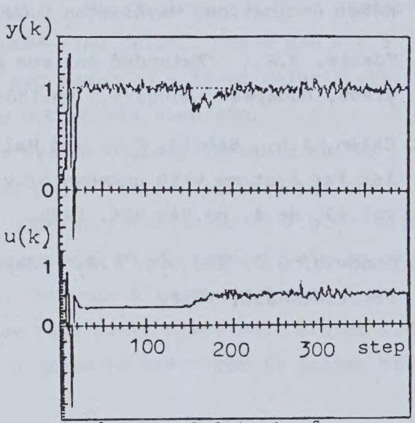


Fig.6 Input & Output of
PASTC-TDE, SNR=28 dB

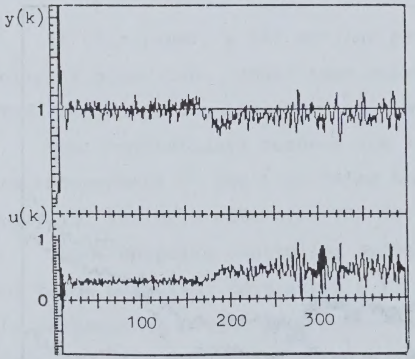


Fig.4 STC-TDE for constant
reference, SNR=24.9 dB

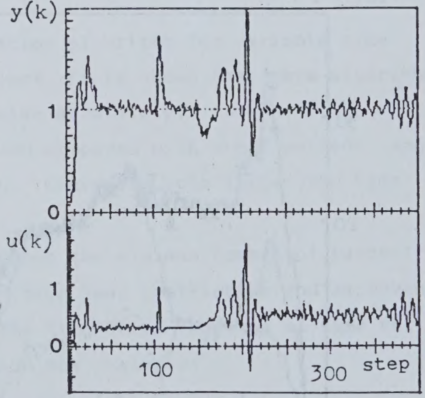


Fig.7 Input & Output
of PASTC, SNR=28 dB

PARAMETER SPACE ANALYSIS OF AN ADAPTIVE SYSTEM

FATHI M. A. SALAM

Department of Electrical Engineering and Systems Science
Michigan State University
East Lansing, Michigan 48824

ABSTRACT

We investigate the possible dynamics that a simple adaptive system may generate in the presence of disturbances. Limiting attention to first order plants with a single unknown pole and an external disturbance, we verify some of the bifurcations that a simulated example has been shown to exhibit. We construct the parameter space, or bifurcation, diagram which summarizes all possible behaviors of the simple adaptive system. We then demonstrate the use of the parameter space as a design tool for the proper utilization of the adaptation scheme.

1. INTRODUCTION

We focus on the so-called σ-modification scheme, introduced by Ioannou and Kokotovic [1], for the design of Model Reference Adaptive Control (MRAC) systems. We are particularly interested in its performance in the face of external disturbances [2,3,1]. As an initial step to a better understanding, the disturbance has been postulated to be constant [2,3]. We note however that even though a simple modification of the scheme would nullify the impact of the constant disturbance, the significance of the formulation and its subsequent analysis is to quantify the capacity of the adaptive scheme to counter unmodeled effects. In addition, it would suggest the performance of the overall system when the disturbance is weakly time varying.

We are concerned with characterizing all the qualitative features- both stabilizing and destabilizing. The relative values of the constant parameters (e.g., unknown poles, unknown disturbances and σ itself), are very crucial in securing a structurally stable adaptive control strategy.

The justification of such a view is that (the sufficient) conditions for ensuring the stability features are often conservative. Moreover, one does not know what combination of values of constant parameters, which satisfy the sufficient conditions, would result in the "most" robust design.

One knows that the σ-modification scheme is associated with the existence of a bounded attracting region in the state space, see Ioannou and Kokotovic [1]. Some viable questions are then: "When would this attracting region contain a single limit set, various limit sets, or infinite limit sets? When would the limit sets entail single or multiple equilibria, periodic orbits, and/or nonperiodic ones?"

To illucidate the proposed viewpoint, we have limited our attention here to the class of first order plants with an unknown pole and external input or output disturbances. Employing a combination of analytical and computational techniques, we have verified, and then generalized [5], the computer simulation results of a simple adaptive system which was reported in [3]. This adaptive system example was first studied by Riedle et. al. [2].

The study of the adaptive system eventually comes down to studying the following 2-d differential equations (see [3])

$$\dot{e} = \rho e - ke + \delta \qquad (1.1i)$$
$$\dot{k} = \sigma k + \gamma e^2, \qquad (1.1ii)$$

where ρ is the unknown pole of the plant, δ is the external disturbance to the plant, σ and γ are adaptation parameters which are to be "properly" chosen by the control designer. The parameters ρ, δ, σ, and γ are nonzero constant.

2. THE ADAPTIVE SYSTEM

Consider the simple adaptive control system:

$$\text{model:} \quad \dot{y}_m = -y_m + r \qquad (2.1)$$

$$\text{plant:} \quad \dot{y} = \rho y + u + \delta \qquad (2.2)$$

$$\text{input:} \quad u = -ky + r \qquad (2.3)$$

$$\text{adaptive law:} \quad \dot{k} = \gamma y(y - y_m) - \sigma k, \quad \gamma > 0, \ \sigma \geq 0. \qquad (2.4)$$

The term $(-\sigma k)$ in the adaptation law was introduced to counteract the presence of the disturbance, δ, in the equation of the plant, see [1,2]. The term $(-\sigma k)$ is referred to as the σ-modification, see Ioannou and Kokotovic [1].

Now let (the error) $e := y - y_m$. Then obtain the equivalent system:

$$\dot{e} = \rho e - ke + \delta + (\rho+1) y_m - k y_m \qquad (2.5)$$

$$\dot{k} = -\sigma k + \gamma e^2 + \gamma e y_m \qquad (2.6)$$

$$\dot{y}_m = -y_m + r \qquad (2.7)$$

As in [2, 3], we are interested in analyzing this simple adaptive system in the regulation case; that is, $r=0$. Also let $\sigma > 0$.

Now, the equilibria must satisfy ([2],[3])

$$k = \rho + (\delta/e), \quad k = (\gamma/\sigma)e^2, \quad y_m = 0. \tag{2.8}$$

Eliminating k in the first two equalities results in the single cubic ploynomial in the variable e,

$$e^3 + (-\rho\sigma/\gamma)\, e + (-\delta\sigma/\gamma) = 0. \tag{2.9}$$

The solution of this polynomial may now be looked up in standard textbooks on Algebra, see [4]. To simplify the represenation of the solutions, we first define the parameters
$a = (-\rho\sigma/\gamma)$, and $b = (-\delta\sigma/\gamma)$,
then we define the quantities

$$A = [-(b/2) + [\,(b/2)^2 + (a/3)^3\,]^{1/2}\ \,]^{1/3},$$

$$B = [-(b/2) - [\,(b/2)^2 + (a/3)^3\,]^{1/2}\ \,]^{1/3}.$$

The three solutions (i.e. roots) of (2.9) may now be formally written as

$$e = A+B, \quad -\tfrac{1}{2}[(A+B) + j\sqrt{3}\,(A-B)], \quad -\tfrac{1}{2}[(A+B) - j\sqrt{3}\,(A-B)]. \tag{2.10}$$

The number of real solutions, however, depends on the sign of the expression

$$\mathcal{H} = (b/2)^2 + (a/3)^3 = (\delta\sigma/2\gamma)\,[\,1 - (4/27)\,(\sigma\rho^3/\gamma\delta^2)];$$

If $\mathcal{H} > 0$, there will be one real solution;
If $\mathcal{H} = 0$, there will be three real solutions two of which are equal; and
If $\mathcal{H} < 0$, there will be three real and distinct solutions.

For simplicity of presentation, we assume that the parameters σ, ρ, γ, and δ are all positive. Then,

$\mathcal{H} > 0$ if and only if $(\delta^2\gamma/\rho^3\sigma) - (4/27) > 0$
$\mathcal{H} = 0$ if and only if $(\delta^2\gamma/\rho^3\sigma) - (4/27) = 0$
$\mathcal{H} < 0$ if and only if $(\delta^2\gamma/\rho^3\sigma) - (4/27) < 0$.

Because of its importance in conducting the analysis, we define the (equilibria bifurcation) expression, which is a mapping from R^4 to R, as

$$\mathcal{H}_b := (\delta^2\gamma/\rho^3\sigma) - (4/27). \tag{2.11}$$

We observe that eqn. (2.7) is linear and is in fact independent from (2.5)-(2.6). Consequently, we may assume that the model has reached its steady state. Thus $y_m(t)=0$ for all t. In that case, the equations one seeks to analyze simplify to

$$\dot{e} = \rho e - ke + \delta \qquad (2.12i)$$

$$\dot{k} = -\sigma k + \gamma e^2 \qquad (2.12ii).$$

Simulation has been conducted on equations (2.5)-(2.7) for $y_m(0)=0$, and thus $y_m(t)=0$ for all t, and various initial conditions of the states e and k. The simulation has been carried out for the fixed parameters:

$\rho = 1.5,\ \gamma = 1.0,\ \sigma = 0.5,$

while δ is varied between 0 and 0.6. The essential and qualitatively different phase portraits have been reported in [3].

More importantly, simulation has also been conducted for fixed ρ and γ at, respectively, 1.5 and 1.0, while varying the two parameters σ and δ. A δ verses σ parameter-space diagram has been constructed with (bifurcation) curves separating various regions. These regions, as well as the curves, identify qualitatively different phase portraits of system (2.12). Such regions have been characterized by identifying the "critical" curves in the (δ,σ)-parameter space on which a bifurcation occurs. These critical curves form the boundaries of the various regions with qualitatively different phase portrait properties. (A parameter space diagram is refered to in the dymanical systems literature as a "bifurcation diagram.")

Figure 1 depicts the (σ,δ)-parameter space diagram. The (bifurcation) curve SN identifies the occurrence of the saddle node bifurcation. It is obtained when the bifurcation expression for equilibria, namely eqn. (2.11), equals zero i.e. when two of the three equilibria become identical. The curve H identifies the occurrence of the (subcritical) Hopf bifurcation. The curve SC identifies the occurrence of the saddle connection, or the homoclinic orbit. While the curve SN was plotted using the expression (2.11), the curves H and SC were plotted by obtaining several points (σ,δ) for which there corresponds a computer-simulated degenrate phase portrait of the system (2.12). The exact numerical values of these points have been determined subjectively via the visual examination of the computer-simulated phase portraits.

3. THE ANALYSIS OF THE ADAPTIVE SYSTEM

We desire to analyze system (2.12)-(2.13), i.e.,

$$e = \rho e - ke + \delta \qquad (3.1i)$$

$k = -\sigma k + \gamma e^2$ (3.1ii).

In this section, we assume that the constant parameters ρ and δ are positive. In order to facilitate the analysis, we introduce the following scaling transformation.

$x := (\rho/\delta)\ e.$ (3.2i)

$y := (1/\rho)\ k.$ (3.2ii)

$\tau := \rho\ t.$
(3.2iii)

$\mu := (\gamma/\rho^4)\ \delta^2.$ (3.2iv)

$\nu := -\ (1/\rho)\ \sigma.$ (3.2v)

The transformation (3.2) transforms system (3.1) to

$\dot{x} = x - xy + 1$ (3.3i)

$\dot{y} = \nu\ y + \mu\ x^2$
(3.3ii).

System (3.3) has only two parameters, ν and μ.

3.1 The equilibria

The equilibria of system (3.3) must simultaneously satisfy the equalities

$y = 1 + (1/x), \quad y = -\ (\mu/\nu)\ x^2.$ (3.4i)

Observe that the first equality, which defines a hyperbola, is independent of the parameters ν and μ. The second equality, which defines a parabola, depends on the ratio (μ/ν) of these parameters. Using (3.4i), we obtain the cubic polynomial in the variable x

$(\mu/\nu)\ x^3 + x + 1 = 0.$ (3.4ii)

In the notation of section 2, assuming for now that ν, $\mu \neq 0$, we have

$a = b = (\nu/\mu)$, and

$A, B = (\nu/2\mu)^{1/3}\ [\ -\ 1 \pm [1 + (4\ \nu/27\mu)]^{1/2}\ ]^{1/3}.$ (3.5)

The solutions, i.e. roots, of (3.4ii), which define the equilibria, are now obtained by substituting A and B of (3.5) into eqn. (2.10). The (equilibria bifurcation) expression is now

$$\mathcal{H}_b := (\mu/\nu) + (4/27). \tag{3.6}$$

Thus for (μ/ν) <, = and >, $-(4/27)$, there exists one, two, and three equilibria, respectively. Note that for a fixed ratio (μ/ν), the location of equilibria is also fixed.

In addition to the ν and μ axes, we employ the point (0,0) and the two half lines with slope $(\mu/\nu) = -(4/27)$ to partition the (ν,μ)-parameter space into six regions.

Corresponding to each region, or line, in the (ν,μ)-parameter space, the curves (3.4i) divide the whole state space (x,y) into a fixed number of regions within which the signs of the derivative quantities $\dot{x}$ and $\dot{y}$ do not change. Now, carefully and patiently, one can sketch the trajectories of the phase portraits in a straightforward manner. We have done so for all the regions 1 through 9 in the parameter space depicted in Figure 2a. However, the phase portrait is not completely determined in all cases. For instance, one cannot conclude the presence or absence of periodic orbits which may surround stable foci for constant parameters lying in region 9 [4,5].

4. PROOF OF THE POINCARE-ANDRONOV-HOPF BIFURCATION: THE GENERATION OF LIMIT CYCLES

Consider region 7 in the parameter space of Figure 2a. Here there coexist three equilibrium points: a stable focus sf, a saddle s, and a focus f. Figure 3 depicts the three equilibria and the qualitative phase portrait of system (3.3). Note the focus f changes stability as the parameter ν decreases from 0. To see this, let the coordinate of the equilibrium f be (x_0, y_0). Then, calculate the characteristic polynomial of the linearization of (3.3) at (x_0, y_0). One thus obtains

$$\phi(\lambda) = \lambda^2 - (1-y_0+\nu)\,\lambda + \nu(1-3y_0) = 0.$$

The component y_0 is always < 1. Hence, for the trace, i.e. $(1-y_0+\nu)$, to change sign, ν must decrease from 0. A necessary condition for the birth of the limit cycle is that the eigenvalues of $\phi(\lambda)$ become pure imaginary and nonzero. Thus,

$$(1-y_0+\nu) = 0, \tag{4.2a}$$

and

$$\nu(1-3y_0) \neq 0. \tag{4.2b}$$

Using (4.2a), (4.2b) becomes

$-\nu(2 + 3\nu) \neq 0.$ (4.2c)

Since ν is negative in region 7, (4.2c) is equivalent to

$\nu \neq 0$ and $\nu \neq -(2/3)$. (4.2d)

Now recall that equilibria must simultaneously satisfy

$y_0 = 1 + (1/x_0)$ and $y_0 = -(\mu/\nu)\, x_0^2$. (4.2e)

The first equality gives $x_0 = (1/\nu)$, and the second gives the following equation for the bifurcation curve for limit cycles.

$\mu = -\nu^3(1+\nu).$ (4.3)

For $\nu \in (-2/3, 0)$, this equation defines the bifurcation of a limit cycle from the focus point f. However, for $\nu \in (-1, -2/3)$, it defines the curve on which the trace of the saddle equals zero [4,5].
Note that the real part of the eigenvalues when they are complex is

$\alpha(\nu) = \frac{1}{2}(1 - y_0 + \nu).$ (4.4)

In order to prove the existence of limit cycle bifurcation, one has to ensure that [6,7]

$(d/d\nu)\, \alpha(\nu)|_{\nu_c} \neq 0.$

From (4.4), we compute

$(d/d\nu)\, \alpha(\nu)|_{\nu_c} = \frac{1}{2}(1 - (d/d\nu) y_0(\nu)|_{\nu_c}).$ (4.5a)

However from eqn. (4.2e) and (4.3), one obtains

$y_0^3 - 2y_0^2 + y_0 = \nu^2(1+\nu).$

Now computing the derivative with respect to ν, one obtains ($y_0 = 1+\nu$)

$(d/d\nu) y_0(\nu)|_{\nu_c} = (2\nu + 3\nu^2) / (3\nu^2 + 2 + 4\nu)$ (4.5b)

Over $\nu \in (-2/3, 0)$, (4.5b) is strictly negative. Consequently, from (4.5a), we conclude that

$(d/d\nu)\, \alpha(\nu)|_{\nu_c} > 0.$

Thus a limit cycle bifurcation necessarily occurs.

To determine the stability of the generated limit cycle, we determine the stability of the equilibrium point f at criticality. Then, the stability of the generated limit cycle would be the same as the stability of f. Along the limit cycle bifurcation curve, the system equation may be expressed as (see [4], [5])

$$\dot{x} = -\nu x - (1/\nu)y - xy \tag{4.6a}$$

$$\dot{y} = (2\mu/\nu)x + \nu y + \mu x^2 \tag{4.6b}$$

Denote the eigenvalues at criticality by $- j\omega$. We have transformed system (4.6) to

$$\dot{x} = -\omega y + f(x,y) \tag{4.7a}$$
$$\dot{y} = \omega x + g(x,y) \tag{4.7b}$$

where

$$f(x,y) = \nu^2 x^2 - \omega\nu xy \tag{4.8a}$$
$$g(x,y) = -\{\nu^2 x^2 + \omega\nu^2 xy\}/\omega \tag{4.8b}$$

Now applying the formula in ([7], eqn. (3.4.11)), one obtains

$$16\, a = 2\nu^2 [-\nu + \nu^2(2+\nu)/\omega^2] > 0. \tag{4.9}$$

Thus the equilibrium f at criticality is (weakly) unstable. Consequently, the bifurcation is subcritical resulting in the generation of an unstable limit cycle.

REFERENCES

[1] P.I. Ioannou and P.V. Kokotovic, "Instability analysis and robustness of adaptive control," Automatica, vol. 20, 1984.

[2] B. Riedle, B. Cyr, and P.V. Kokotovic, "Disturbance Instability in an Adaptive System," IEEE Trans. on Automatic Control, vol. AC-29, No. 9, September 1984, pp. 822-824.

[3] F.M.A. Salam and S. Bai,"Disturbance-Generated Bifurcations in a Simple Adaptive System: Simulation Evidence," Systems and Control Letters, vol. 7, No. 4, 1986.

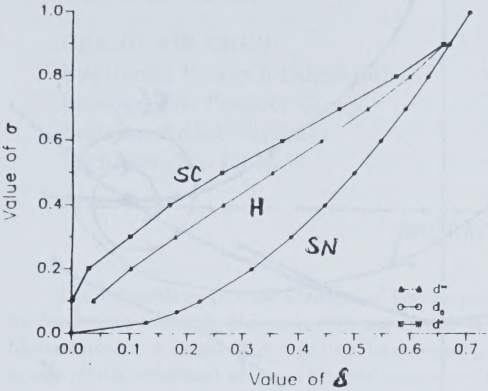
[4] F.M.A. Salam,"Parameter Space Analysis and Design For Model Reference Adaptive Control Systems," IEEE CAS (in press).

[5] F.M.A. Salam and Z. Zhifen, "Bifurcation Analysis of a Simple Adaptive System," SIAM on Mathematical Analysis (in press).

[6] S.N. Chow and J.K. Hale, Methods of Bifurcation Theory, Springer-Verlag, 1982.

[7] J. Guckenheimer and P. Holmes, Nonlinear Oscillations, and Dynamical Systems and Bifurcations of Vector Fields, Springer-Verlag,1983.

FIGURE CAPTIONS



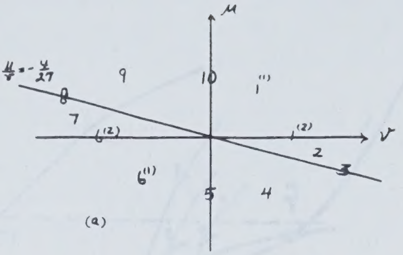

μ
ν
μ/ν = -ν/27
1
1(1)
1(2)
2
3
4
5
6(1)
6(2)
7
8
9
10
(a)

Figure 2a.

Figure 1. Parameter space diagram for the parameters δ and σ: simulation only.

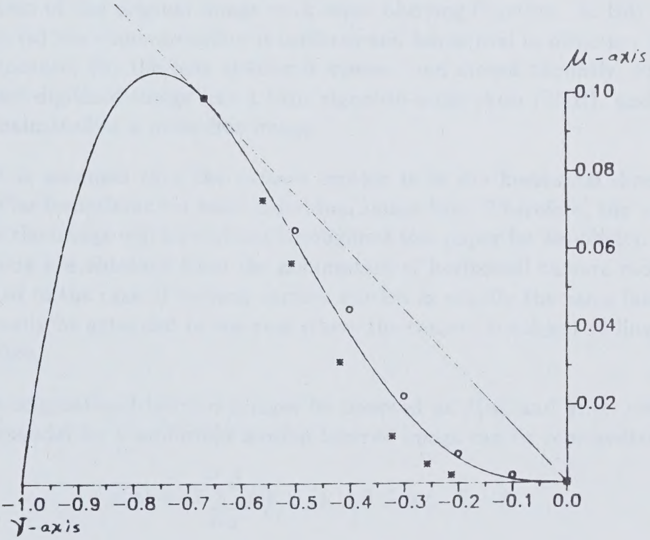


Figure 2b. Parameter space diagram for the parameters ν and μ: simulation.

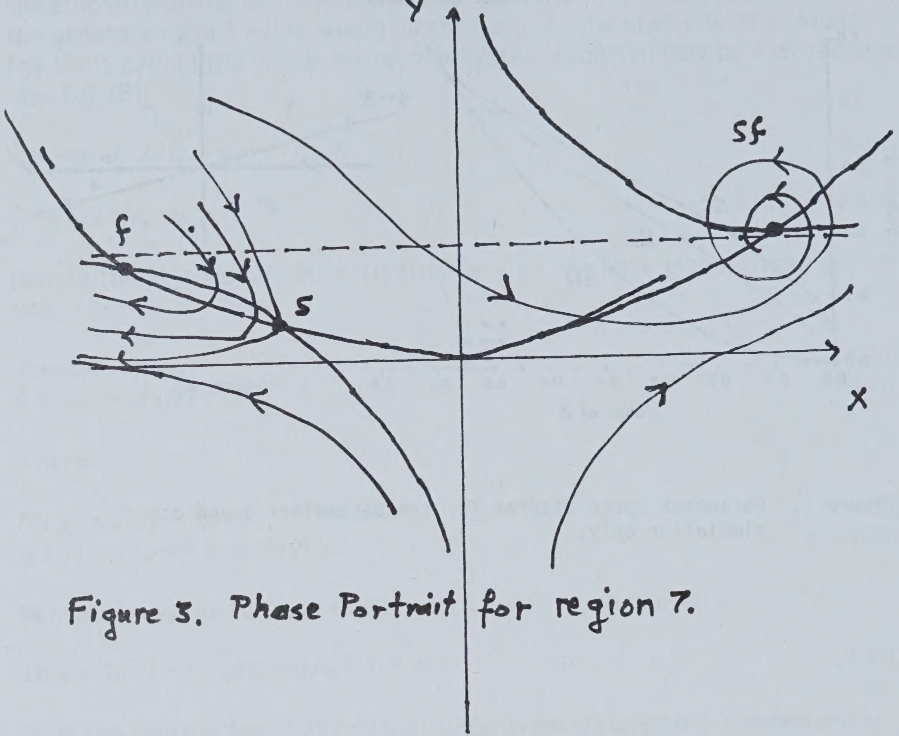


Figure 3. Phase Portrait for region 7.

RESTORATION OF SMEARED IMAGES SUBJECT TO CAMERA MOTION

CHENG-TIE CHEN
Electronics Research Laboratories -
Photographic Product Group
Eastman Kodak Company
Rochester, NY 14650

ABSTRACT

The restoration process consists of two steps. First, a new method has been developed to determine correctly the unknown parameter in the blurring function from the truncated blurred image. A linear filter with the determined parameter is then used to deblur the smeared image. Both simulated and real blurred images are considered in the examples to illustrate the restoration procedure. Good restored images are obtained.

I. INTRODUCTION

Although today's camera technology is so advanced that the lens can focus on the object automatically, it is still not advanced enough to correct image degradation caused by camera motion automatically. In this paper, we consider a postexposure restoration method to deblur the smeared image caused by camera motion.

Generally speaking, the blurred image can be modeled as the sum of noise and the convolution of the original image with some blurring function. In this paper, we assume that: (a) the camera motion is uniform and horizontal in direction during the period of exposure; (b) the lens shutter is opened and closed abruptly; and (c) the developed and digitized image has a high signal-to-noise ratio (SNR), and therefore can be approximated as a noise-free image.

Since it is assumed that the camera motion is in the horizontal direction, the problem can be formulated for each individual image line. Therefore, the vertical coordinates for the image will be omitted throughout this paper for simplicity. Although the results here are obtained from the assumption of horizontal camera motion, they can be applied to the case of vertical camera motion in exactly the same fashion. The results can easily be extended to the case where the camera is subject to linear motion in any direction.

Let the original and blurred images be denoted as $f(n)$ and $g(n)$, respectively. The discrete model for a uniformly motion-blurred image can be represented by

$$g(n) = C \sum_{i=0}^{D-1} f(n-i), \quad 0 \leq n \leq N-1, \tag{1}$$

where N is the number of elements per line, D stands for the displacement of the camera motion, and C is some positive constant to normalize the exposure. Note that $g(n)$ is the convolution of $f(n)$ and some blurring function $h(n)$, where $h(n)$ is a rectangular pulse function.

There are two parameters involved in the blurring function, pulse magnitude C and pulse width D. The pulse magnitude reflecting the film exposure is just a normalizing constant and, therefore, is not important. However, the pulse width D, a quantity representing the displacement of the camera motion, is critical in the restoration process.

Several approaches to finding the blurring function have been proposed [1,2]. Since the blurring function under consideration is a simple rectangular pulse (where only the width needs to be determined), those sophisticated approaches are not suitable because of the noninvertibility of the blurring function and the bit-precision limitation. It has been proposed [3] that the displacement D can be determined from the nulls of the magnitude spectrum of the blurred image by taking advantage of the fact that the frequency response of the rectangular pulse is a sinc function vanishing at $\pm 2\pi/D$, $\pm 4\pi/D$, etc. In theory, there is nothing wrong with this method. However, in practice, the blurred image boundaries are always truncated, owing to the finite film window. Does this method still work for this windowed blurred image? *Generally*, the answer is "No." This motivated us to investigate further.

In Section II, we review briefly the conventional method in determining the motion displacement D. In Section III, we describe a new method developed to determine the motion displacement D. The boundary truncation has essentially no effect on the correct determination of D with this method. The idea is to consider some function of the blurred image, instead of the blurred image itself. Such a function eliminates the boundary effects because the function values at the truncated boundaries are so small that they can be approximated as zeros. Also, the function shows some inherent characteristic, which enables us to determine the motion displacement D correctly.

After D has been determined, the next step is to deblur the degraded image with the appropriate filters. The filter considered in this paper is the constrained minimum mean-square (CMMS) filter [4]. The extrapolation approach in [5] is adopted at the boundaries to lessen the ringing effects. The CMMS filter with extrapolation is briefly summarized in Section IV.

Examples are given in Section V to illustrate the method discussed above. First, the computer-simulated blurred image is considered so that the performance superiority of our new method over the conventional method [3] can be justified. Finally, a real blurred image exposed on the film is considered. Good performance was observed with our method for the real blurred image.

II. REVIEW OF CONVENTIONAL METHOD

Let $F(\omega)$ and $G(\omega)$ denote the Fourier transforms (FT) of $f(n)$ and $g(n)$, respectively. From Eq. (1), it is straightforward to show that

$$|G(\omega)| = |F(\omega)| \times |H(\omega)|, \tag{2}$$

where $|H(\omega)|$, the magnitude of the FT of the blurring function $h(n)$, is given by

$$|H(\omega)| = C|\sin(\omega D/2)|/|\sin(\omega/2)|. \tag{3}$$

Taking a logarithm on both sides of Eq. (2), we obtain

$$\log|G(\omega)| = \log|F(\omega)| + \log|H(\omega)|. \tag{4}$$

Note that $|H(\omega)|$ vanishes periodically at $\omega = \pm 2\pi/D, \pm 4\pi/D, \cdots$, etc. Therefore, periodic nulls should be observed in $\log|G(\omega)|$ (see Eq. (4)). In theory, we should be able to determine the quantity D from the positions of these nulls.

Note that the positions of these nulls depend only on D, regardless of image data. Therefore, it is wise to average $|G(\omega)|$ over all image lines so that the undesired nulls, which may occur in some particular image lines, will not dominate over the desired ones. One inherent advantage of averaging $|G(\omega)|$ is that the noise (e.g., 8-bit quantization noise) perturbation in the blurred image can be reduced. Two averaging processes — arithmetic mean and geometric mean — are considered. The version of Eq. (4) obtained by using the arithmetic mean of $|G(\omega)|$ over all image lines is given by

$$\log\{|G(\omega)|\}_{ave} = \log|H(\omega)| + \log\{|F(\omega)|\}_{ave}, \tag{5}$$

where $\{x\}_{ave}$ stands for the sum of all possible x divided by the number of terms in that summation. Similarly, for the geometric mean, we have

$$\{\log|G(\omega)|\}_{ave} = \log|H(\omega)| + \{\log|F(\omega)|\}_{ave}. \tag{6}$$

Experimental results show that Eqs. (5) and (6) give substantially the same results. However, Eq. (5) is computationally more efficient. Note that $\log|H(\omega)|$ appears in both Eqs. (5) and (6). Therefore, the periodic nulls, whose locations depend on D only, should be observed after $|G(\omega)|$ is averaged.

In theory, the above scheme has nothing wrong. However, in practice, the blurred images are truncated at the boundaries, to some extent. This may occur because the image is developed only within the film aperture. For those astronomical images, the backgrounds are black usually. A small truncation at the boundaries does not result in any loss of information. However, for most customer images, the backgrounds are so dynamic that even a small truncation causes a loss of information. The loss of information is equivalent to introducing zeros in $|G(\omega)|$, which will interfere with the desired zeros. Particularly, the FT of the actual blurred image is the convolution of a sinc function and the FT of the original image for rectangular film aperture. Therefore, such a boundary truncation results in some ambiguity of null locations in $\log|G(\omega)|$, and makes the determination of D totally impossible. This will be illustrated in Section V.

The boundary truncation effects can be lessened by manipulating the blurred image data before taking the FT. One commonly used method [5] is to extrapolate the data in such a way that the values gradually fall off to zero. In Section V, examples will be given to evaluate the performance of this scheme. It will be seen that we still suffer from some difficulties even if we extrapolate the blurred data.

In the next section, we will propose a new method to determine the motion displacement D. The boundary truncation has essentially no effect on the determination of D for this method.

III. A NEW METHOD TO DETERMINE MOTION DISPLACEMENT

This new method does not consider the blurred-image data directly. Instead, we define a new image data $g'(n)$ as the difference of every two adjacent pixels of the blurred image. Mathematically,

$$g'(n) = g(n) - g(n-1), \quad 1 \leq n \leq N-1. \tag{7}$$

Some straightforward manipulations, after substituting Eq. (1) into Eq. (7), give

$$g'(n) = C\{f(n) - f(n-D)\}. \tag{8}$$

Let $G'(\omega)$ denote the FT of $g'(n)$. Taking the magnitude of the FT on both sides of Eq. (8) gives

$$|G'(\omega)| = |F(\omega)| \times |H'(\omega)|, \tag{9}$$

where

$$|H'(\omega)| = C|\sin(\omega D/2)|. \tag{10}$$

Since $|H'(\omega)|$ vanishes at $\omega = 0, \pm 2\pi/D, \pm 4\pi/D$, etc., there will be nulls occurring at the same locations in $\log|G'(\omega)|$.

As in Section II, arithmetic and geometric means can be considered for $|G'(\omega)|$ to reduce possible dominant effects from some particular image lines. The corresponding versions of Eqs. (5) and (6) are given by

$$\log\{|G'(\omega)|\}_{ave} = \log|H'(\omega)| + \log\{|F(\omega)|\}_{ave} \tag{11}$$

and

$$\{\log|G'(\omega)|\}_{ave} = \log|H'(\omega)| + \{\log|F(\omega)|\}_{ave}, \tag{12}$$

respectively. Experimental results show no substantial difference between Eqs. (11) and (12). However, Eq. (11) is preferrable because of its computational effficiency.

It will be seen in Section V that this new method out-performs the conventional one described in Section II. Note that the boundary truncation also introduces some loss of information in $g'(n)$. Then, why does this new method not experience the problems encountered in the conventional method? The reason is as follows. Consider a nontruncated blurred image generated from a finite-support original sharp image (see Eq. (1)). Obviously, the pixel values at the boundaries should increase or decrease slowly. The data $g'(n)$, defined as the difference of every two adjacent blurred pixels, should be very small in the nontruncated case. Therefore, such a boundary truncation in $g'(n)$ introduces only small and tolerable errors in $|G'(\omega)|$.

IV. DEBLURRING (RESTORATION) FILTER

The deblurring filter considered in this paper is the constrained minimum mean-square (CMMS) filter [4]. The CMMS filter is obtained by minimzing the power of the restored image subject to the constraint that the input noise power is known. It can be shown that the CMMS filter is equal to the Wiener filter when the power spectral density ratio between signal and noise remains constant for all frequencies. The particular CMMS filter $H_0(\omega)$ is given by

$$H_0(\omega) = H^\star(\omega)/\{|H(\omega)|^2 + \epsilon\}, \tag{13}$$

where $\star$ stands for complex conjugate and $H(\omega)$ represents the FT of the blurring function $h(n)$. The ϵ in Eq. (13) is a small positive constant related to the input noise power. A further investigation on the application of the CMMS filters can be found in [5].

It has been shown [5] that boundary truncation introduces objectionable ringing artifacts in the restored image. To reduce these artifacts, we will use an extrapolation scheme [5] to smooth the data before the CMMS filter is applied.

V. RESULTS

The size of the original sharp image in Fig. 1 is 500×500 ($N_0 = 500$) in red, green, and blue. Each pixel is quantized into 8 bits. We simulated a blurred image by convolving the original with a rectangular pulse function of width $D = 25$. The pulse amplitude C is set to be $1/D$ to normalize the image intensity. After convolution, the pixel values are quantized into 8 bits by using a particular calibration table to simulate the scanner. Finally, the image is truncated at the boundaries to simulate the environment of the finite film aperture. The number of elements per line N for the blurred image under consideration is $N_0 - D + 1$. The resulting blurred image is shown in Fig. 2. Note that there is quantization noise involved although the noise-free approximation assumption has been made. This will give us some insight into the robustness of the estimation methods to the noise perturbation, which is practically unavoidable. The averaged magnitude spectrum in this section refers to Eqs. (5) and (11), i.e., the arithmetic mean.

The averaged $|G(\omega)|$ in dB for the blurred image in Fig. 2 is shown in Fig. 3(a). Here a 1024-point fast Fourier transform (FFT) was used. We would expect that the locations of the nulls should be at the frequency samples 41, 82, $\cdots$, etc., for $D = 25$. Although the values at these frequency samples do get very small, boundary-truncation effects make the determination of these locations totally impossible, as seen in Fig. 3(a).

As noted in Section II, an extrapolation scheme can be used to reduce the boundary-truncation effect. Before taking the FT, the blurred data are extrapolated at the left and right boundaries so that the values fall off gradually to zero. The averaged $|G(\omega)|$ in dB are given in Figs. 3(b) and 3(c) for the extrapolated blurred image, where 25- and 40-point linear extrapolations at both left and right boundaries are considered, respectively. Obviously, these curves are better than Fig. 3(a). In fact, 25-point extrapolation gives satisfactory results. However, we suffer from some ambiguities in determining the null loactions for the case of 40-point extrapolation. Apparently, the disavantage of this scheme is that the performance depends on the number of extrapolation points.

The averaged $|G'(\omega)|$ in dB for the blurred image in Fig. 2 is shown in Fig. 4. It is expected, from the results in Section III, that the nulls should occur at the frequency samples 0, 41, 82, $\cdots$, etc. Indeed, from the curve in Fig. 4, the locations of the nulls can be unambiguously and correctly determined. Without doubt, the new method gives us much more satisfactory results for the truncated blurred image.

We now apply the CMMS filter to the blurred image in Fig. 2. A 25-point linear extrapolation at the left and right boundaries is used to reduce the ringing artifacts. The restored image with $\epsilon = -40$ dB is given in Fig. 5. It can seen that the sharpness has been restored. There are still some artifacts in the restored image, visible especially in the smooth areas. The causes of these artifacts are investigated in detail in [5].

Shown in Fig. 6 is a blurred image taken on KODACOLOR VR 200 Film. The pixel resolution is about 20 microns and each pixel is quantized into 8 bits through a calibration table. The image size is 800×240 in red (R), green (G), and blue (B). Note that there are film noise (e.g., grain noise) and other film parameter perturbations involved in the blurred data, in addition to the 8-bit quantization noise. The averaged $|G(\omega)|$ and $|G'(\omega)|$ are shown in Figs. 7 and 8, respectively, where a 2048-point FFT is used. The luminance data Y (=0.3R+0.59G+0.11B) are considered here. Although only the first null is recognizable, the performance of the averaged $|G'(\omega)|$ is still much better in determining the location of the null. The location of the first null is estimated to be at frequency sample 116, which corresponds to $D = 2048/116 \approx 18$.

Applying the CMMS filter ($D = 18$) with $\epsilon = -20$ dB and 18-point linear extrapolation at the left and right boundaries of the blurred image gives the restored image in Fig. 9. It can be seen that the sharpness has been restored; some characters smeared in the blurred image now become clearly recognizable.

VI. SUMMARY AND CONCLUSION

We have developed a new method to determine the parameter in the blurring function. It has been demonstrated that this new method performs much better than the conventional method. The CMMS filter with extrapolation was used to deblur smeared images after the parameter was determined. Examples were considered for both simulated and true blurred images. In both cases, good results were obtained.

Blurring function identification is a difficult problem, considering the perturbations in real-life situations. Also, deblurring smeared images is a poorly-posed problem if the blurring function is not invertible. The author feels that without having other *a priori* knowledge such a problem cannot be solved perfectly. More research on those aspects needs to be done.

REFERENCES

[1] H. Kaufman, J. W. Woods, S. Dravida, and A. M. Tekalp, "Estimation and identification of Two-Dimensional Images," *IEEE Trans. Automatic Control*, Vol. 28, pp. 745-756, 1983.

[2] A. M. Tekalp, H. Kaufman, and J. W. Woods, "Identification of Image and Blur Parameters for Restoration of Noncausal Blurs," *Proceedings IEEE International Conf. ASSP 85*, pp. 656-659, 1985.

[3] D. Slepian, "Restoration of Photographs Blurred by Image Motion," *Bell Syst. Tech. J.*, Vol. 46, pp. 2353-2362, 1967.

[4] H. C. Andrew and B. R. Hunt, *Digital Image Restoration*, Prentice-Hall: Englewood Cliffs NJ, 1977.

[5] C.-T. Chen, "An Investigation to Reduce Ringing Artifacts in Image Restoration," to be published.

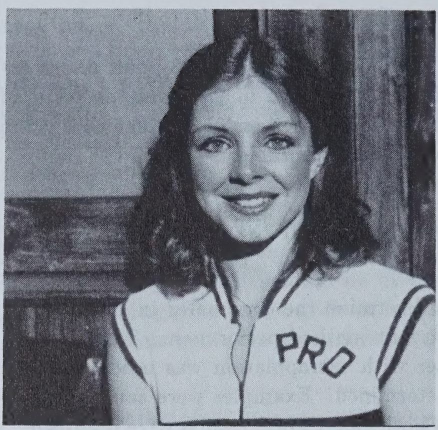


Figure 1: Original sharp image (500 elements per line and 500 lines).

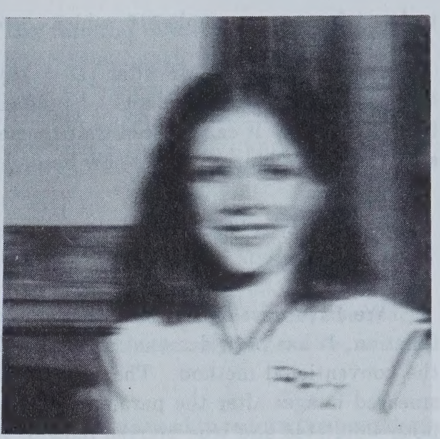


Figure 2: Blurred image (476 elements per line and 500 lines) with D=25.

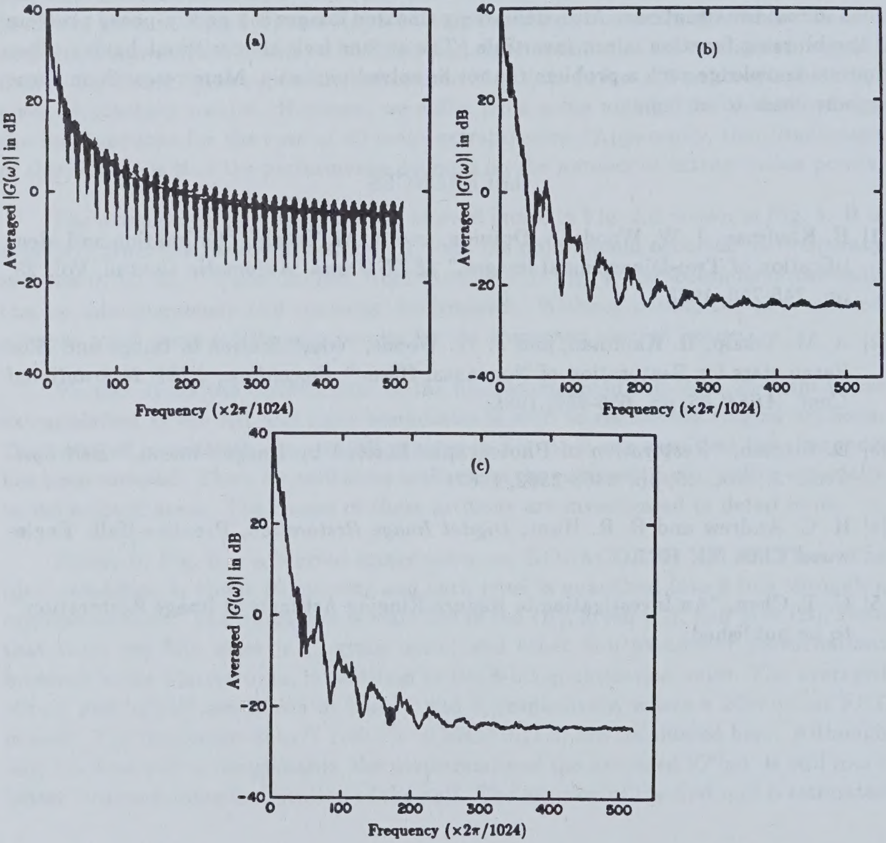


Figure 3: Averaged $|G(\omega)|$ in dB for Fig. 2: (a) no extrapolation; (b) 25-point extrapolation; (c) 40-point extrapolation.

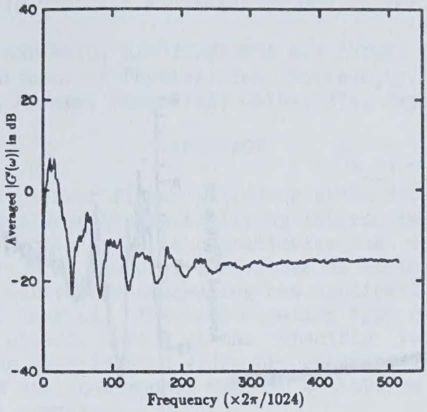


Figure 4: Averaged $|G'(\omega)|$ in dB for Fig. 2.

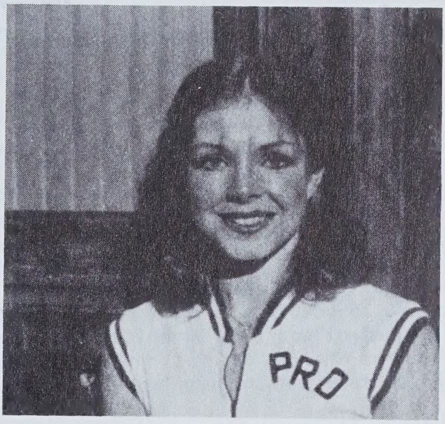


Figure 5: Restored image from Fig. 2 with $D = 25$, $\epsilon = -40$ dB, and 25-point extrapolation.

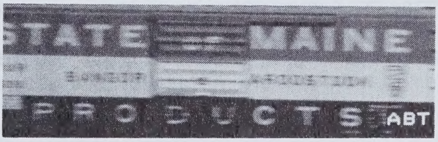

STATE
MAINE
PRODUCTS
ABT

Figure 6: Blurred image taken on KODACOLOR VR 200 Film; 800 elements per line and 240 lines.

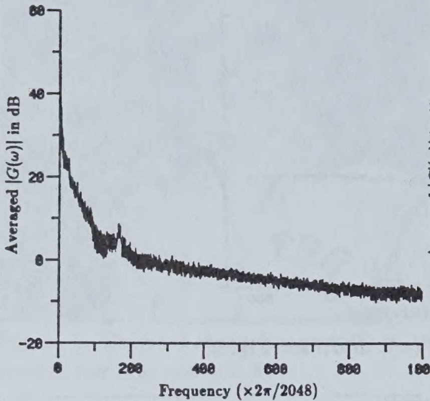


Figure 7: Averaged $|G(\omega)|$ in dB for Fig. 6.

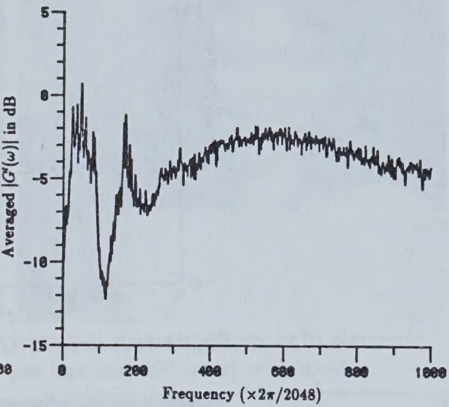


Figure 8: Averaged $|G'(\omega)|$ in dB for Fig. 6.

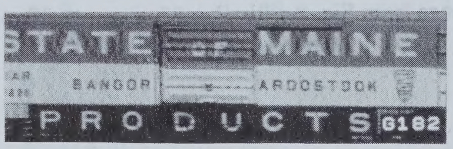

STATE OF MAINE
BANGOR
AROOSTOOK
PRODUCTS 6182

Figure 9: Restored image from Fig. 6 with $D = 18$, $\epsilon = -20$ dB, and 18-point extrapolation.

SPATIO-TEMPORAL FILTERING OF MOVING IMAGE

SHUJI HASHIMOTO, HIROYOSHI ODA and YOICHI SATO
Department of Physics, Toho University,
2-2-1, Miyama, Funabashi, Chiba, 274, Japan

ABSTRACT

Spatio-temporal linear filter including up to second derivatives on time and space are analyzed theoretically by introducing not only frequency space but also velocity space to characterize the operator effects on moving image. The role of temporal derivative on moving targets is similar to that of spatial derivative suggesting new applications on moving image processing and 3-D imaging. The wave equation type recursive filter is proved to detect objects moving at the specific velocity. The system architecture of the experimental video processor is also given which is designed to perform spatio-temporal feature extraction on real time by the analyzed 3x3x3 local operators.

I. INTRODUCTION

Recently, the moving image became a target of image processing in various applications such as narrow band television, trucking and monitoring systems, as high speed memories and logic circuits are developed[1]-[3]. However, theoretical analyses of moving image filtering have not been sufficiently done in the view of dynamic feature extraction, and most of algorithms are discovered heuristically. One possible way to understand moving image is to repeat still image processing at frame by frame to trace back on feature changing. However, especially in pre-processing, we consider that it is important to treat moving image as a dynamic scalar field and to express filters by partial differential equations. Because mathematical analyses are rather simple and prospective to advance general theory of moving image processing. For example, movement detection has been performed by the comparison of two images on different frames, which should be considered as a spatio-temporal derivative of the time varying image[4]. Moreover, these filters can be easily realized for digital image using spatio-temporal local operators to make the system simple for high speed operation.

In this paper, characteristics of spatio-temporal linear operator up to the second derivatives are discussed first, by introducing the wave number-velocity space instead of the Fourier space where conventional filter characterizations are done[5]. Second, the effect of temporal derivative on the object image moving at constant velocity is discussed and the basic ideas of velocity-dependent image extractions are proposed. Finally, the architecture of our experimental real time video processor is given, which performs the discussed spatio-temporal local operations.

II. FILTER REPRESENTATION IN WAVE NUMBER-VELOCITY SPACE

Let f(x,t) and g(x,t) be an input image and an output image respectively, as functions of position x and time t. Then the filters under consideration are expressed by the following two types of equations, as

$$g(x,t) = L\{f(x,t)\}, \tag{1}$$

$$L\{g(x,t)\} = f(x,t), \tag{2}$$

where $L\{\ \}$ represents linear operator. Equation (1) is non-recursive filter while Equation (2) is recursive one. In the following discussions, we restrict $L\{\ \}$ to include operators up to the second derivatives in space and time, so that the filters refer only to the nearest neighbor pixels in calculations to make the system simple for the fast operation as described below.

The frequency response function H(k,w) of the filter in wave number-temporal frequency (k-w) space can be obtained by the multi-dimensional Fourier transformation of the above equations as,

$$H(k,w) = \left| \frac{G(k,w)}{F(k,w)} \right| , \tag{3}$$

where F(k,w) and G(k,w) are Fourier transformations of the input f(x,t) and the output image g(x,t), respectively[6]. The wave number response will give image deformation due to the filter, while the temporal frequency response will not, because the dynamic features of image which attract our attentions are not related to the temporal changes of individual pixel intensities, but to the pattern movement and deformation on time, except for special applications such as the blinking area detection. Therefore, we propose the wave number-velocity (k-v) space as a response description space for spatio-temporal image filters. The k-v space response M(k,v) of filter can be easily obtained by assuming that

$$f(x,t) = f(x-vt). \tag{4}$$

The above assumption is fit for the case that solid objects are moving at constant velocity v. If the sinusoidal pattern of wave number k will move along x-axis at velocity v, then the image will have a temporal frequency component kv. Therefore, in the one dimensional case, the k-v response function M(k,v) can be obtained by replacing w in H(k,w) by kv. M(k,v) gives an absolute output intensity for the input pattern of wave number k and velocity v.

The followings are some interesting examples in one dimensional space.

Example 1 : Spatial derivative

$$g(x,t) = \frac{\partial f(x,t)}{\partial x} , \quad H(k,w) = \frac{k\pi}{L} , \quad M(k,v) = \frac{k\pi}{L} , \tag{5}$$

where 2L is the size of image area.

Example 2 : Temporal derivative

$$g(x,t) = \frac{\partial f(x,t)}{\partial t} , \quad H(k,w) = w , \quad M(k,v) = \frac{\pi}{L} kv . \tag{6}$$

The frequency responses of $\partial/\partial x$ and $\partial/\partial t$ are k and w respectively and are quite different. However, k-v responses are similar as shown in Fig.1. The temporal derivative filter will extract detail structures of moving image and suppress still image and lower wave number components to be used for moving edge detection.

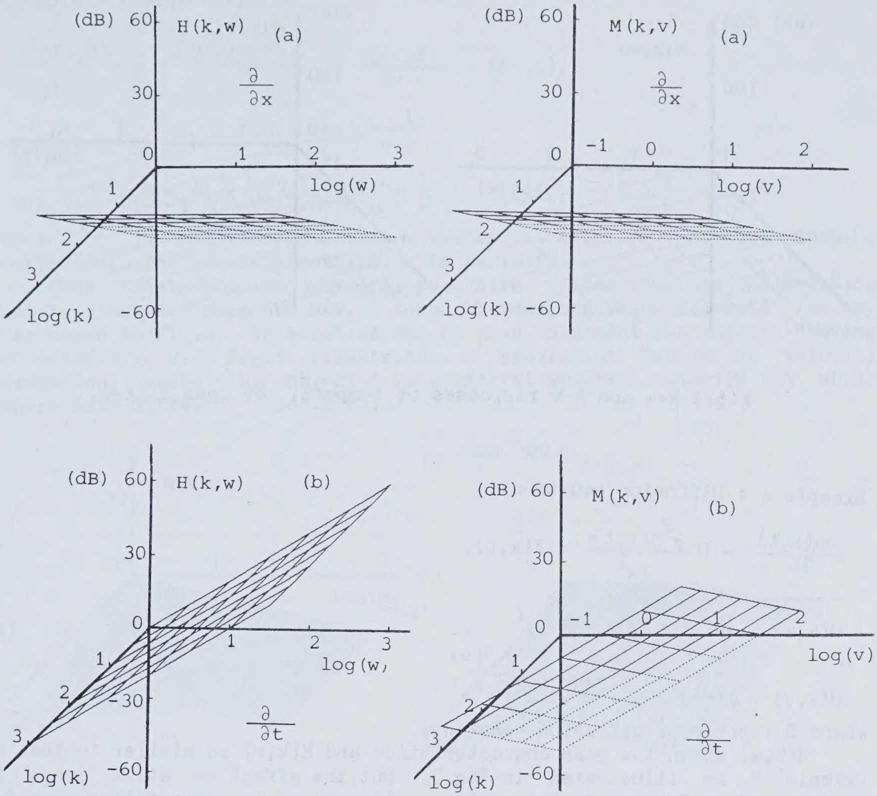


Fig.1 k-w and k-v responses of spatial derivative (a) and temporal derivative (b).

Example 3 : Temporal low pass filter

$$\frac{\partial g(x,t)}{\partial t} + \tau g(x,t) = f(x,t),$$

$$H(k,w) = (w^2+\tau^2)^{-\frac{1}{2}}, \qquad (7)$$

$$M(k,v) = \left\{(\frac{\pi}{L})^2 k^2 v^2 + \tau^2\right\}^{-\frac{1}{2}},$$

where τ represents time constant.

The operator does not depend on space coordinate x, similar to Example 2, and H(k,w) is a function of w only. However, in k-v space , M(k,v) is a function of both k and v as shown in Fig.2. This filter will suppress the detail structures of the moving object image, while will not effect on the still image.

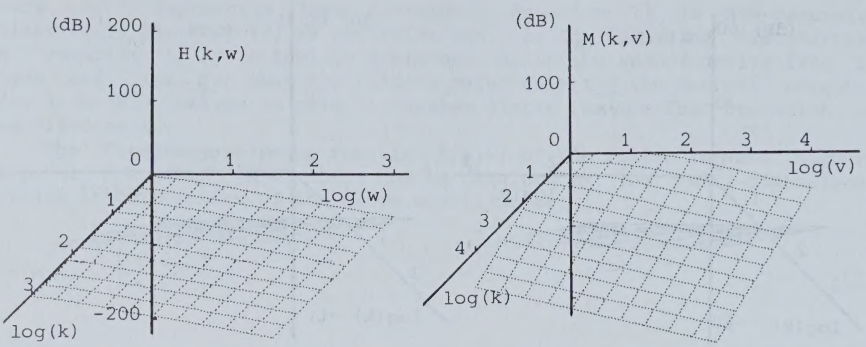


Fig.2 k-w and k-v responses of temporal low pass filter.

Example 4 : Diffusion equation

$$\frac{\partial g(x,t)}{\partial t} - D\frac{\partial^2 g(x,t)}{\partial x^2} = f(x,t),$$

$$H(k,w) = \{w^2+(\frac{k\pi}{L})^4 D^2\}^{-\frac{1}{2}} \tag{8}$$

$$M(k,v) = \{(\frac{\pi}{L})^2 k^2 v^2+(\frac{k\pi}{L})^4 D^2\}^{-\frac{1}{2}}$$

where D represents diffusion constant.

H(k,w) shows low pass characteristics and M(k,v) is similar to that of Example 3 as illustrated in Fig.3. But the effect on still image is different from Example 3, this filter will work as low pass filter even for v=0 to blur the input image.

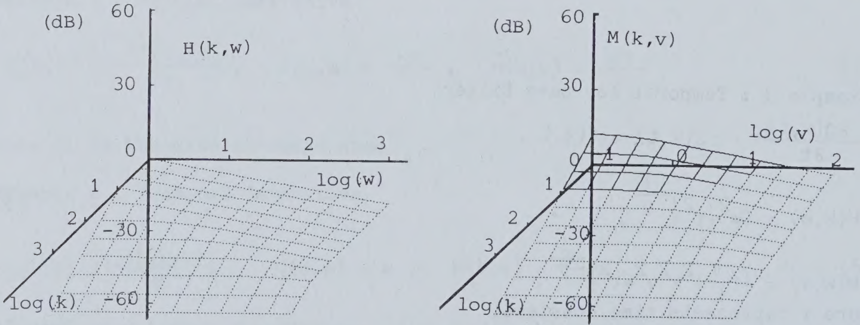


Fig.3 k-w and k-v responses of diffusion equation.

Example 5 : Wave equation

$$\frac{\partial^2 g(x,t)}{\partial t^2} - c^2\frac{\partial^2 g(x,t)}{\partial x^2} - r\frac{\partial g(x,t)}{\partial x} = f(x,t), \tag{9}$$

$$H(k,w) = \left[\left\{\left(\frac{k\pi}{L}\right)c^2 - w^2\right\}^2 + \left(\frac{rk\pi}{L}\right)^2\right]^{-\frac{1}{2}},$$

$$M(k,v) = \left[\left(\frac{k\pi}{L}\right)^4 (c^2 - v^2)^2 + \left(\frac{rk\pi}{L}\right)^2\right]^{-\frac{1}{2}},$$

where C is the characteristic velocity and r is the dumping coefficient. The boundary conditions is periodic.

This most popular physical recursive filter has an interesting feature that k-v response M(k,v) takes the maximum value at v = C for any k as shown in Fig.4. This filter can be used to detect the objects moving at velocity C. Fig.5 illustrates a simulation result of velocity extraction, where the object A is moving at specific velocity C=1 while others have different velocities.

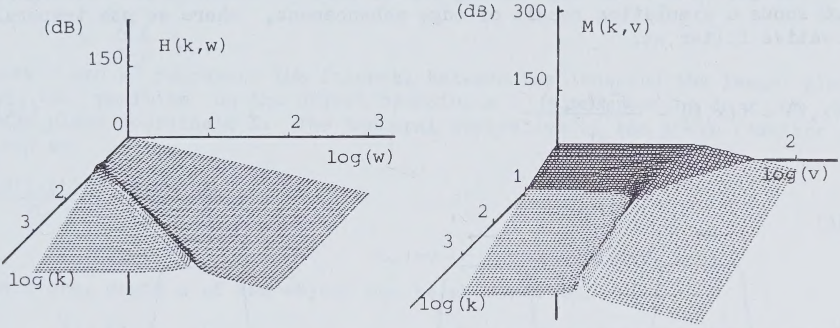


Fig.4 k-w and k-v responses of wave equation.

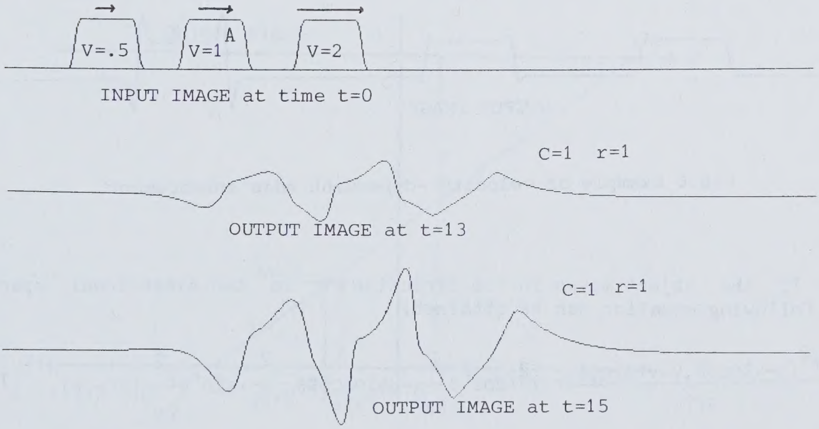


Fig.5 Input and output images of wave equation type filter.

III. APPLICATION OF TEMPORAL DERIVATIVE

As mentioned above, the temporal derivative filter responds as spatial derivative one to moving images, which can be proved mathematically as follows

$$\frac{\partial f(x-vt)}{\partial t} = f' \frac{d(x-vt)}{dt} = -vf' . \tag{10}$$

In the same manner, for the second derivative,

$$\frac{\partial^2 f(x-vt)}{\partial t^2} = v^2 f'' \tag{11}$$

These results suggests that by using temporal derivatives in stead of spatial derivatives, we can perform the velocity-dependent edge enhancement of the image, where the edge moving faster will be enhanced more strongly. Fig.6 shows a simulation result of edge enhancement, where we use temporal derivative filter as,

$$g(x,t) = f(x,t) - \frac{\partial^2 f(x,t)}{\partial t^2} \tag{12}$$

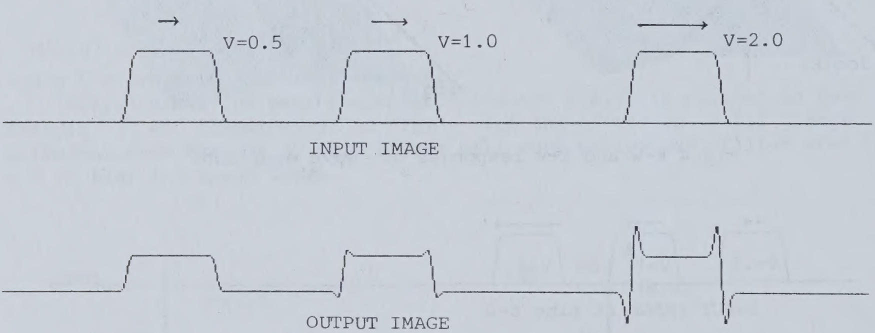


Fig.6 Example of velocity -dependent edge enhancement.

If the object moves in the direction ϑ in two dimensional space, the following equation can be obtained.

$$\frac{\partial f^2(x-vt\cos\vartheta, y-vt\sin\vartheta)}{\partial t^2} = v^2\left\{\cos^2\vartheta\frac{\partial^2}{\partial x^2} - 2\sin\vartheta\cos\vartheta\frac{\partial^2}{\partial x\partial y} + \sin^2\vartheta\frac{\partial^2}{\partial y^2}\right\} f(x,t). \tag{13}$$

In the case that the camera facing a still image is vibrating in all the directions, the mean output g(x,y) of the second temporal derivative filter is given by averaging the above equation with respect to $(0<\vartheta<2\pi)$ as

$$g(x,y) = \frac{v^2}{2}\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)f(x,y). \tag{14}$$

Therefore, it is found that the second derivative on time axis accompanied with the camera vibration will work as Laplacian filter for two dimensional still image.

Moreover, if the image is the projection of 3-dimensional still scene, the temporal output, while camera is vibrating, will give the depth information of the scene, because the apparent movement of the object is the function of the distance between the camera and the object as follows. Let e(t) be the displacement of camera axis O from the object axis O' at time t, and z be the distance between object plane and the image plane of the camera.(See Fig.7.) Then the object image f(x,t) obtained on camera plane can be expressed as,

$$\begin{aligned} f(x,t) &= f\left(x+\frac{r}{z-r}e(t)\right), \\ x &= \frac{x'r}{z-r}, \end{aligned} \tag{15}$$

where r and x' represent the interval between the lens and the image plane and the position on the object coordinate X' corresponding to x on the image plane coordinate X. The temporal derivative of the above function is given as,

$$\frac{\partial f(x,t)}{\partial t} = f'\frac{r}{z-r}\frac{de}{dt}, \tag{16}$$

Therefore, depth z of the object can be measured as,

$$z = \frac{f'r\frac{de}{dt}}{\frac{\partial f}{\partial t}} + r. \tag{17}$$

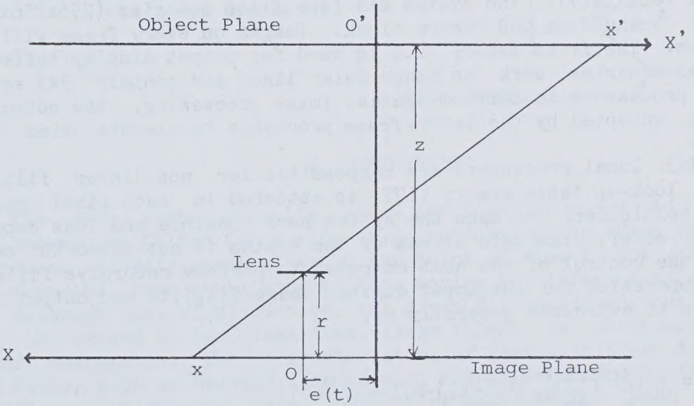


Fig.7 Relation between camera displacement and apparent object image.

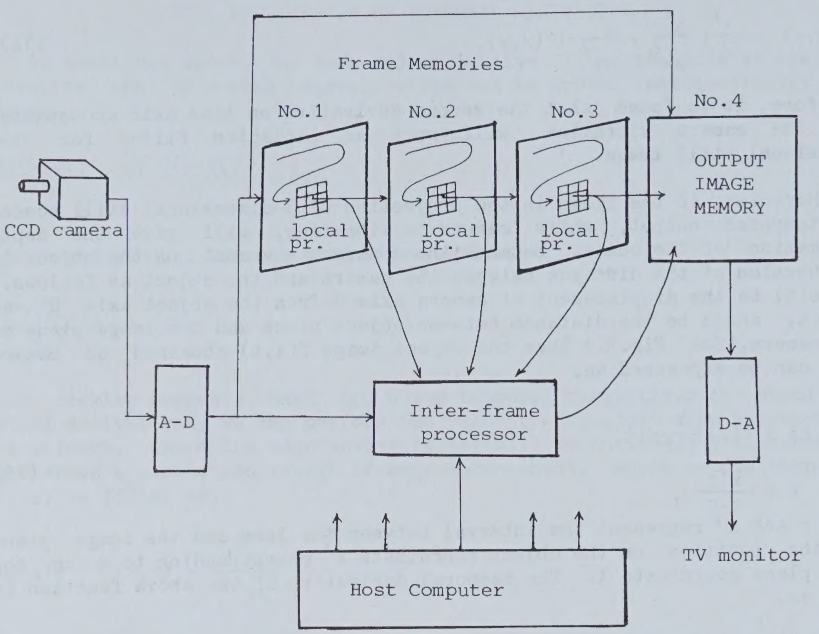


Fig.8 Brief block diagram of the experimental video processor.

IV. EXPERIMENTAL REAL TIME VIDEO PROCESSOR (EXVP-1)

All the spatio-temporal filtering described here can be realized by 3x3x3 spatial temporal cubic convolutor for digital images. Fig.8 is the block diagram of the experimental system EXVP-1 constructed to apply the theoretical results[7]. The system has four frame memories (256x256x8 bit capacities) scanned on CCD camera clock. Images on every frame will pass through them. The frame memory No.4 is used for output display buffer. The other frame memories work as image delay line and contain 3x3 scanning local area processors to perform spatial image processing, the outputs of which are accepted by the inter-frame processor to execute time domain processing.

The 3x3 local processors are responsible for non-linear filtering, because a look-up table memory (LUT) is attached to each pixel register instead of multiplier, to make the system more flexible and less expensive (Fig.9). Moreover, the data stream of the system is not fixed but can be changed by the control of the host computer to perform recursive filtering. The total operation for the input digital image f(i,j,t) and output image g(i,j,t) can be expressed generally as,

$$g(i,j,t) = T_0\{\sum_k T^k[\sum_\ell\sum_m T^k_{\ell m}\{f(i-\ell,j-m,t-k)\}]\} \qquad \text{(nonrecursive)}, \quad (18)$$

$$g(i,j,t) = T_0\{\sum_k T^k[\sum_{\ell m}\sum T^k_{\ell m}\{g(i-\ell,j-m,t-k)\}]\} + f(i,j,t) \quad \text{(recursive)}, \tag{19}$$

where T^k 'S and T_0 are arbitrary functions given by LUT's in k-th frame local processor and in the inter-frame processor.
Each LUT memory can store eight tables simultaneously to change the filter characteristics on real time for time dependent processing and adaptive filtering. The system can be successfully operated at rate of 200ns/pixel for real time video processing.

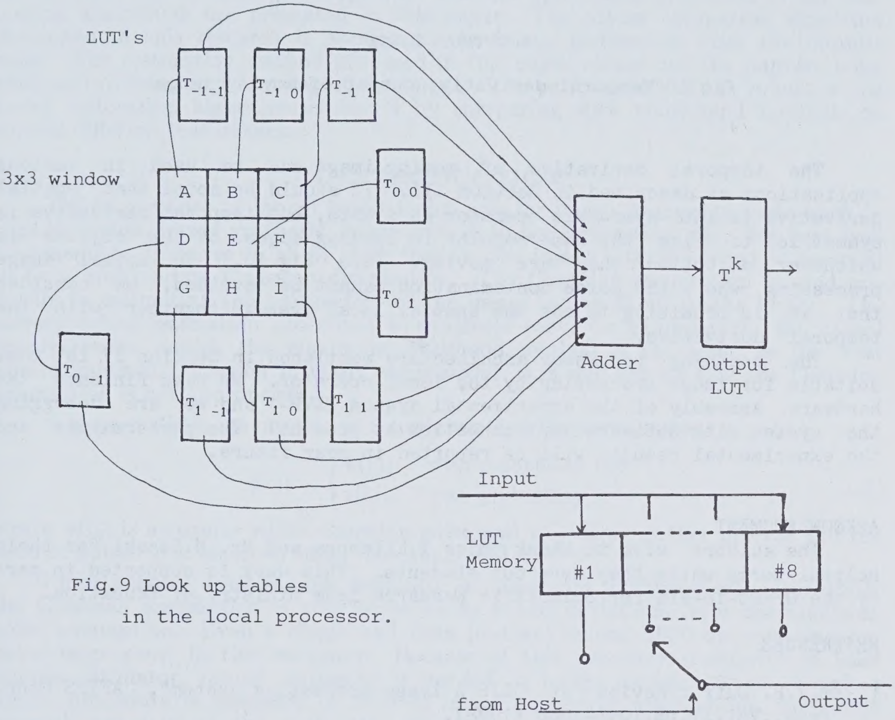


Fig.9 Look up tables in the local processor.

V. CONCLUSIONS

The wave number-velocity space has been proposed to characterize spatio-temporal filter for moving image processing. As shown in Section II, introduced velocity space gives us new point of view : How will the filter transform the image containing objects moving at different velocities ?

Although most of discussions are given for one-dimensional case, it is easy to expand to two dimensional image filter by considering velocity vector components. The characteristics of partial differential equation type filter such as Example 4 and 5 will change according to their boundary conditions, which may have interesting applications such as position dependent feature extraction. We did not discussed about the stability of given recursive filters in this paper, because it is almost same to the numerical solution stability of partial differential equations.

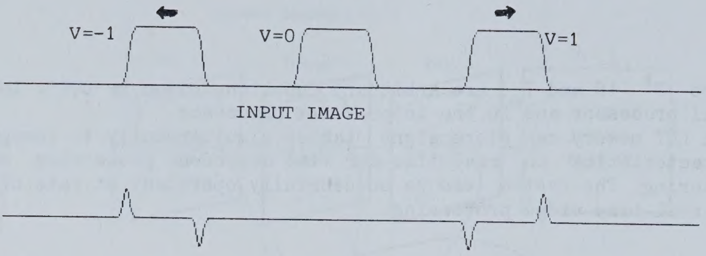


Fig.10 Temporal derivative output of moving image.

The temporal derivative of moving image can be used in various applications as described in Section III. It should be noted that spatial derivative is anti-symmetric operator on x axis, but temporal derivative is symmetric to give the same results to leading edges of the objects in whichever direction they are moving. (See Fig.10.) In actual image processing where the noise contamination cannot be avoided, we consider that it is promising to use the spatial local average together with the temporal derivative.

The scanning processor architecture mentioned in Section IV is most suitable for image processing by the local operator. We have finished the hardware assembly of the experimental system EXVP-1 and we are debugging the system with software implementation at present. The performances and the experimental results will be reported in near future.

ACKNOWLEDGMENT

The authors wish to thank Miss Y.Kitamura and Mr. M.Suzuki for their helpful works while they were our students. This work is supported in part by the Grant-in-Aid for Scientific Research from Ministry of Education.

REFERENCES

[1] M.J.B. Duff :"Review of CLIP 4 image processing system", AFIPS Conf. Proc.,Vol.47,pp.1055-1060 (1978).
[2] D.K. Arvind, I.N. Robinson and I.N. Parker :"A VLSI chip for real-time image processing", IEEE Int. Symp. Circuits System, pp.405-408 (1983).
[3] R. M. Lougheed, D. L. McCubbrey : "The Cytocomputer : A Practical pipelined image processor", Pro. 7-th Annual Int. Simposium on Computer Architecture,(1980).
[4] S. Sabri and B. Prasada :"Video conferencing systems", Proc. IEEE, Vol. 73 (1985).
[5] L.T. Bruton and N.R. Bartley: "Three-dimensional Image Processing Using The Concept of Network Resonance", IEEE Trans. Circuits Syst., Vol.CAS-32, pp.664-672(1985).
[6] P.K. Rajan : "A Study on the properties of multi-dimensional Fourier transforms", IEEE Trans. Circuits Syst., Vol.CAS-31, pp.748-750(1984).
[7] S. Hashimoto, Y. Sato and H. Oda: "Real-Time Video Processor with Spatio-Temporal Local Operator", Proc. Annual Convention of IECE JAPAN, 1304(1986).

ROBUST IMAGE MODELLING TECHNIQUES WITH AN IMAGE RESTORATION APPLICATION†

KIE-BUM EOM
Dept. of Electrical and
Computer Engineering
Syracuse University
Syracuse, New York 13244

R. L. KASHYAP
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

ABSTRACT

Robust estimation methods based on M-estimator are applied to two types of outlier problems in causal autoregressive models. Theoretical properties of the estimation algorithms are presented in this paper. The robust estimation algorithm developed in this research is applied to the image restoration from the impulse noise. The restoration method proposed in this paper cleans out the impulse noise without involving any blurring of the image. The performance of the robust model based restoration algorithm is tested by comparing with traditional methods on several different real images.

I. INTRODUCTION

The importance of model based techniques for image processing tasks such as edge detection, image synthesis, image coding, image restoration, etc. has been well documented. However in all of these models, the image intensity array is assumed to be a multivariate Gaussian distribution. The Gaussian assumption is used primarily in estimating the parameters of the image model fitted to the image. The corresponding estimation procedure is relatively easy; for example, for the causal autoregressive model, the maximum likelihood method is the same as the least squares method. However in many applications, it is well known that the Gaussian assumption is not appropriate.

A more realistic assumption is a contaminated Gaussian noise.

$$\zeta(i,j) = \begin{cases} w(i,j), & \text{with probability } 1-\beta \\ v(i,j), & \text{with probability } \beta \end{cases} \tag{1.1}$$

where w(i,j) is a regular white Gaussian noise and v(i,j) is an outlier process and the ratio of outlier β is assumed small (less than 5%).

Unfortunately, least squares estimator or maximum likelihood estimator under the Gaussian assumption is very sensitive to minor deviations from the Gaussian noise assumption. Even a single bad data (outlier) among 1000 observations can cause large error in the estimator. Because of this excessive sensitivity of least squares estimator, robust estimator is needed in image models. Even though a robust procedure is necessary in most of image processing applications, very little research was done on the application of robust procedure in image processing. In this paper, we develop estimation algorithms for causal autoregressive image model and apply this robust method to image restoration problem.

Restoration of an image in the presence of noise is one of the fundamental problems in image processing. As we discussed in the beginning of this section, the Gaussian assumption of noise process is not realistic and the noise process should be included outliers (impulse noise). There are many image restoration methods based on the Gaussian noise assumption. For instance, Chellappa and Kashyap [11] used spatial interaction models to represent image intensity array and restored images with minimum mean square error criterion. Unfortunately, most image restoration methods based on the Gaussian noise assumption are not effective in the presence of impulse noise [1]. The impulsive component of the noise, which is also called as salt and pepper noise, is only a small portion (usually less than 5%) of the total image but difficult to be removed by the methods based on the Gaussian noise assumption, because its amplitude is much higher than signal amplitude.

---

† Partially supported by the Office of Naval Research under the grant N00014-85K-0611 and by the National Science Foundation under the grant IST 8405052

The importance of this problem has been recognized for a long period of time. Traditionally, nonlinear filtering methods such as median filter [1] or α-trimmed mean filter [6] are used to remove impulse noise from the image. These traditional restoration methods assume that image intensity in a small window is a constant and replace the intensity at center pixel by its location estimate. However in these traditional methods, the edges are often smeared and overall output image is often blurred. A good example of blurring effect of median filter can be found in [1].

Of course, there has been some approaches to reduce these blurring effect of the traditional impulse noise removing methods. For instance, Yasuoka and Haralick used a facet model based approach[10], Kundu et. al.[12] used a generalized mean filter. However, these methods are not robust to impulse noise. We develop a restoration method which can remove impulse noise and does not smear the edges and does not blur the image after restoration. Our method uses causal autoregressive model to represent image intensity array and the robust estimation procedure is applied to remove noise.

The restoration algorithm based on the robust image modelling approach is tested with several real images. The experiment shows that the new approach based on the robust modelling gives better results than traditional methods such as median filter or α-trimmed mean filter in the presence of impulse noise.

II. CAUSAL AUTOREGRESSIVE MODEL

Suppose that $\{\varsigma(i,j)\}$ is a two dimensional white noise sequence with outliers as assumed in (1.1). The variance of the regular part of noise is σ^2. Then the three neighbor causal autoregressive model is represented by the following equation.

$$y(i,j) = \theta^T z(i,j) + \varsigma(i,j) \tag{2.1}$$

where $\boldsymbol{\theta}$ is a parameter vector and z(i,j) is a vector consists of intensities of three causal neighbors and unity. The last element of the vector z(i,j) is used to represent constant grey level in the image.

$$z(i,j) = \begin{bmatrix} y(i,j-1) \\ y(i-1,j) \\ y(i-1,j-1) \\ 1 \end{bmatrix} \tag{2.2}$$

It is assumed that every pixel has all of its neighbors, i.e., for each pixel at (i,j), pixels at (i,j-1), (i-1,j) and (i-1,j-1) are available.

We consider the robust parameter estimation of the causal autoregressive model for two cases of outliers. First case, we assume that the process y(i,j) given in (2.1) can be prefectly observed. In this case, the outlier process is involved only in the noise process $\varsigma(i,j)$ to generate y(i,j). Second case, we assume that the observation x(i,j) of the process y(i,j) is corrupted by noise $\xi(i,j)$. It is given by the following equation.

$$x(i,j) = y(i,j) + \xi(i,j) \tag{2.3}$$

The noise process ξ is assumed containing outliers. In this case, the outliers are not only involved in generating y(i,j) but also involved in observation. In the next section, robust parameter estimation will be discussed for these two different cases of outliers.

III. ROBUST PARAMETER ESTIMATION

A. Perfect Observation Case

The parameters of image model given in (2.1) can be estimated by robust M-estimator. The M-estimator of the parameters in (2.1) is a generalization of location M-estimator. Define the following function $Q(\theta,\sigma)$.

$$Q(\theta,\sigma) = \frac{1}{mn}\sum\left[\rho\left(\frac{y(i,j) - \theta^T z(i,j)}{\sigma}\right) + \frac{1}{2}\right]\sigma \tag{3.1}$$

where ρ is a continuous, differentiable and convex function possessing bounded derivative, and it is symmetric about origin with $\rho(0) = 0$. Then M-estimator of causal autoregressive model is defined by the following minimization problem.

$$\text{Minimize } Q(\theta,\sigma) \tag{3.2}$$

The M-estimator can also be obtained by solving the following two equations simultaneously.

$$\nabla_\theta Q(\theta,\sigma) = \frac{-1}{mn}\sum\psi\left(\frac{y(i,j)-\theta^T z(i,j)}{\sigma}\right)z^T(i,j) = 0 \tag{3.3}$$

$$\frac{\partial Q(\theta,\sigma)}{\partial\sigma} = \frac{1}{2} - \frac{1}{mn}\sum\chi\left(\frac{y(i,j) - \theta^T z(i,j)}{\sigma}\right) = 0 \tag{3.4}$$

where $\psi(x)=\frac{\partial\rho(x)}{\partial x}$ and $\chi(x)=x\psi(x)-\rho(x)$, function ψ is continuous and bounded. The following ρ, and ψ functions satisfy above conditions on these functions. In this section, it is assumed that the following functions are used in our robust estimation algorithm.

$$\rho(x) = \begin{cases} \frac{1}{2}x^2, & |x| \leq c \\ c\,|x| - \frac{1}{2}c^2, & |x| \geq c \end{cases} \tag{3.5}$$

$$\psi(x) = \frac{d\rho(x)}{dx} = \begin{cases} c, & x \geq c \\ x, & -c \leq x \leq c \\ -c, & x \leq -c \end{cases} \tag{3.6}$$

Asymptotic Property: The asymptotic property of the robust M-estimator for autoregression is investigated by Nasburg and Kashyap [9]. The asymptotic property of one dimensional autoregression is also applicable to two dimensional causal autoregressive model. Now define $\hat{\theta}_N$ be an M-estimator which satisfies (3.2) and computed with sample size N. Under some assumptions given in [9], the parameter estimates $\hat{\theta}_N$ is consistent, and converges to a zero mean Gaussian distribution.

Computation of the Estimator: The robust M-estimators defined by (3.2) can be computed by the following iterative algorithm.

Algorithm 1 (Estimation with Perfect Observation):

1. Let $\theta^{(0)}$, and $\sigma^{(0)}$ be initial estimates (conventional least square estimator can be used).
2. At k-th iteration, Compute the following quantities by the given formula.

$$r^{(k)}(i,j) = y(i,j) - \theta^{(k)T} z^{(k)}(i,j) \tag{3.7}$$

$$\hat{r}^{(k)}(i,j) = \psi\left(\frac{r^{(k)}(i,j)}{\sigma^{(k)}}\right)\sigma^{(k)} \tag{3.8}$$

$$\tau^{(k)} = [\sum_{i,j} z(i,j)z^T(i,j)]^{-1}[\sum_{i,j} z(i,j)\hat{r}^{(k)}(i,j)] \tag{3.9}$$

where z(i,j) is as defined in (2.2).

3. Update parameters by the following formula.

$$\theta^{(k+1)} = \theta^{(k)} + \tau^{(k)} \tag{3.10}$$

$$\sigma^{(k+1)^2} = \frac{1}{mn}\sum_{i,j}[\hat{r}^{(k)}(i,j)]^2 \tag{3.11}$$

4. Repeat steps 2-3 until the differences $||\theta^{(k+1)}-\theta^{(k)}||$ and $|\sigma^{(k+1)}-\sigma^{(k)}|$ become negligible.

The iterative estimation algorithm (algorithm 1) decreases the value of the function Q in (3.1) as the number of iteration increase and converges to the minimum value of function Q. The estimated parameters also converge to the parameters which minimize the function Q in (3.1). The above property of the algorithm is summarized in the following theorem 1.

Theorem 1: Assume that the ψ function given in (3.6) is used and $\hat{\sigma}>0$, the sequence $\{(\theta^{(k)},\sigma^{(k)})\}$ converges to $(\hat{\theta},\hat{\sigma})$, a unique solution of (3.3) and (3.4) and an estimator which minimize (3.2).

In our experiment, the estimation algorithm 1 converges very fast. In most of cases, the estimates are converged in less than three iterations.

B. Noisy Observation Case

In this section, we assume that the process y(i,j) is not observable and the observation x(i,j) is corrupted by noise with outlier as in (2.3). As we have shown in the introduction, M-estimator is defined by a minimization of a likelihood function type loss function. However, the M-estimator of the problem with noisy observation is difficult to define. Kleiner, Martin, and Thomson [4] considered the following minimization problem for one dimensional autoregression with some heuristic arguments.

$$\text{Minimize} \sum_{i} \rho\left(\frac{x(i) - \theta^T \hat{z}(i)}{\sigma} \right)$$

where $\hat{z}$ is an estimator of state vector z. However, it is not very helpful to use the above criterion because the resulting estimator can be far from robust estimator depending on the state estimate $\hat{z}$.

The estimation with noisy observation can be done by combining the robust estimation algorithm for the perfect observation and data cleaning procedure. In this algorithm, the original data (uncorrupted data) is also estimated at each iteration by a data cleaning procedure. The detailed data cleaning method is given in section IV. At each iteration, the cleaned data at the preceding iteration is used to estimate parameters by algorithm 1 and the data is again cleaned by using the estimated parameters.

Choice of ψ functions: The ψ function is used in both algorithm 1 and data cleaning procedure. The good choice of ψ function is not only important for the robustness of estimator but also important for the fast convergence of the iterative procedure. A hard limiter type ψ function is introduced in (3.6).

Another interesting ψ-function is the following Hampel's ψ-function. The Hampel's function is also continuous but returning to zero outside of some interval. It is known that the redescending ψ function yields higher efficiencies than monotone ψ function for extremely heavy tailed distributions [2]. The above advantage of the redescending ψ function is also confirmed in our experiment: The procedure converges much faster with Hampel's redescending ψ function (3.15) than with Huber's monotone ψ function (3.6). This function performed best with parameters a=2, b=2.5, c=4.5 in our experiment.

$$\psi_{HA}(x) = \begin{cases} x, & |x| \le a \\ \frac{a}{(b-a)}(b-x), & a<x\le b \\ \frac{a}{(b-a)}(b+x), & -b\le x<-a \\ 0, & |x| > b \end{cases} \tag{3.15}$$

IV. APPLICATION TO IMAGE RESTORATION

The purpose of image restoration is to remove noise, including impulse noise, from the image. The image degradation process can be represented by the equation (2.3). In this case, x is the observation, y is the original image intensity, and ξ is the noise process with outlier. Image restoration is an estimation of original intensity y from the observation x. For a small sized image, original image intensity can be modelled by a causal autoregressive model. If the original image intensity follows a causal autoregressive model, then the original image intensity can be easily recovered by the robust estimation algorithm for noisy observation case. The data cleaning procedure removes outliers at each iteration without degrading original signal.

The restoration method based on the robust image model has an advantage over conventional methods such as median filter or α-trimmed mean filter. Robust image model based method does not blur images after restoration. Conventional methods, such as median filter or α-trimmed mean filter, replace every pixel by its location estimates. Because these methods are based on the constant intensity assumption, the details of original image are significantly blurred. This procedure at each iteration is described in the following block diagram (figure 1) and the algorithm is also summarized below.

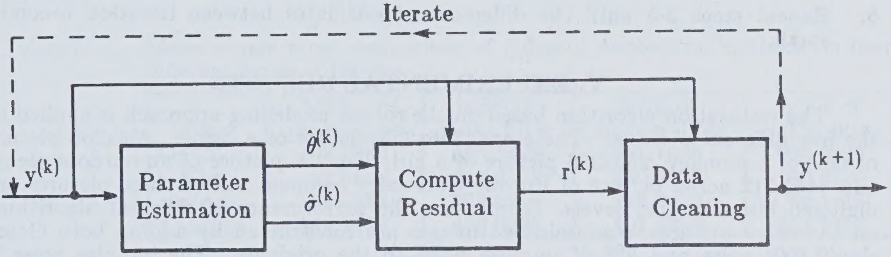


Figure 1. Block diagram of image restoration method at each iteration; $y^{(k)}$ and $y^{(k+1)}$ are cleaned data at k-th and (k+1)-th iterations respectively, $\hat{\theta}^{(k)}$ and $\sigma^{(k)}$ are parameter estimates obtained by algorithm 1, and $r^{(k)}$ is the residual.

Image Restoration Algorithm

1. Divide the image into small sized (8x8) windows and the following procedures in steps 2-6 are applied for each window.
2. Let x(i,j) represents the given noisy data in the window and $y^{(k)}(i,j)$ represents the cleaned data at the k-th iteration. Initially, $y^{(0)}(i,j) = x(i,j)$ for all (i,j). Compute initial estimators $\theta^{(0)}$ and $\sigma^{(0)}$ by least squares method.

$$\theta^{(0)} = [\sum_{i,j} z^{(0)}(i,j) z^{(0)^T}(i,j)]^{-1} [\sum_{i,j} z^{(0)}(i,j) y^{(0)}(i,j)] \tag{4.1}$$

$$\sigma^{(0)^2} = \frac{1}{mn} \sum_{i,j} [y^{(0)}(i,j) - \theta^{(0)^T} z^{(0)}(i,j)]^2 \tag{4.2}$$

where m and n are row and column dimensions of the image and $z^{(k)}(i,j)$ is the following state vector.

$$z^{(k)}(i,j) = \begin{bmatrix} y^{(k)}(i,j-1) \\ y^{(k)}(i-1,j) \\ y^{(k)}(i-1,j-1) \\ 1 \end{bmatrix} \tag{4.3}$$

3. Consider k-th iteration, $k>0$. Compute residuals $r^{(k)}(i,j)$ and modified residual $\hat{r}^{(k)}(i,j)$ by the following formula with the estimated parameters computed in step 2 for all pixels in the window.

$$r^{(k)}(i,j) = y^{(k)}(i,j) - \theta^{(k)T} z^{(k)}(i,j) \tag{4.4}$$

$$\hat{r}^{(k)}(i,j) = \psi\left(\frac{r^{(k)}(i,j)}{\sigma^{(k)}} \right) \sigma^{(k)} \tag{4.5}$$

where ψ is one of the bounded and continuous functions as we discussed in section 3.

4. Restore image by the following rule.

$$y^{(k+1)}(i,j) = \theta^{(k)T} z^{(k)}(i,j) + \hat{r}^{(k)}(i,j) \tag{4.6}$$

5. Update estimators of parameter $\boldsymbol{\theta}$ and scale parameter σ^2 by the following formula.

$$\theta^{(k+1)} = \theta^{(k)} + [\sum_{i,j} z^{(k)}(i,j) z^{(k)T}(i,j)]^{-1} [\sum_{i,j} z^{(k)}(i,j) \hat{r}^{(k)}(i,j)] \tag{4.6}$$

$$\sigma^{(k+1)^2} = \frac{1}{mn} \sum_{i,j} [\hat{r}^{(k)}(i,j)]^2 \tag{4.7}$$

6. Repeat steps 3-5 until the difference of estimates between iteration becomes small.

V. EXPERIMENTAL RESULTS

The restoration algorithm based on the robust modelling approach is applied to the five different pictures. These are 256x256 picture of a bridge, 256x256 picture of face of a monkey, 256x256 picture of a girl, 256x256 picture of an outdoor scene, and 512X512 aerial picture of Purdue University campus. All of these pictures are digitized into 256 grey levels. To measure the performance of different algorithms on the noisy pictures, contaminated images are constructed by adding both Gaussian(0,100) noise and 5% of impulse noise to the originals. The impulse noise is generated to have equal probability of having grey levels 0 (black) and 255 (white). In the robust model based algorithm, Hampel's ψ-function is used in all experiment. Experiments are performed in three different aspects with contaminated pictures. First, the convergence of the restoration algorithm was shown with these noisy pictures, and the rate of convergence is measured experimentally. Second, the mean square error of three different restoration algorithms, model based algorithm, median filter, and α trimmed mean filter, are compared for different window sizes and different images. Third, the overall performance of three different restoration algorithms are compared qualitatively for different noisy images.

A. Convergence of Image Restoration Algorithm

The robust model based restoration algorithm is applied to the contaminated images. Mean square error of the cleaned image is computed at each iteration. The change of mean square error by the progress of iteration is plotted with the number of iterations.

Figures 2 is a typical convergence plot of mean square error with respect to number of iterations from zero to ten. Initial mean square error of all cases are very large because of the additive noise, but they decrease very fast in the first and second iterations. The mean square error converges in less than three iterations. These observations show that the convergence of the data cleaning method is very fast.

B. Mean Square Error Comparison

Four different types of image restoration methods with different size of windows of 3x3, 5x5, and 7x7, are used in this experiment. These are mean filter, median filter, α-trimmed mean filter with the trimming ratio α=0.15, and the robust model based method with window size 8x8. Note that the popular choice of α is in the range from 0.1 to 0.15, and the method performed best with choice α=0.15 in our experiment. In the case of robust model based method, the fixed window size of 8x8 is used. The choice of 8x8 is from the convenience and the small change of window size would not affect the performance, because we used the image model to represent image intensity in the window. The computed mean square error is plotted with respect to window size.

Figures 3 is a typical plot of mean square error computed by different methods. The results are consistent for all different types of images. All traditional methods result in large values of mean square error on most of images especially on the images having many edges. The mean square error comparison shows that robust model based method perform better than any other conventional methods on tested images. Table 1 summarizes the results on mean square error of different methods. Each entry of each method is the minimum value of mean square error. The minimum values of mean square error in conventional methods are 220.2222 for outdoor scene, 80.6720 for girl, 92.1115 for bridge, 253.7658 for Purdue campus, respectively. Mean square errors of our approach are 103.9669 for outdoor scene, 52.5648 for girl, 47.3367 for bridge, 189.1443 for Purdue campus. The level of mean square error of conventional methods is always higher than that of robust model based method.

Table 1. Mean square error comparison of different restoration methods on four different types of images.

Image	MSE of our method	MSE of mean filter	MSE of median filter	MSE of α-TM filter
Outdoor	103.9669	690.1441	651.1638	220.2222
Girl	52.5648	318.9122	300.3172	80.6720
Bridge	47.3367	264.6290	216.3370	92.1115
Campus	189.1433	453.9291	401.6255	253.7658

C. Qualitative Comparison

The noisy image and images restored by different restoration algorithms is shown in figure 4. Figures 4 is the result on the bridge scene. Upper left corner of the figure 4 is the noisy picture contaminated by noise and is generated by adding white Gaussian(0,100) noise and 5% of impulse noise to the original. This image shows typical salt and pepper noise pattern as well as Gaussian noise degradation. This noisy picture is used to obtain restored images by different methods.

Upper right corner of figure 4 is the restored image by robust model based method. This image is obtained after three iterations of data cleaning process. The impulsive noise is almost completely absent, and residual Gaussian noise are hardly noticeable. The fine details of the restored image are very well preserved. As a matter of fact, almost all details of the original is very well shown in this picture. For example, guy wire of the bridge has very sharp edges as in the original. This result shows the important ability of the image model based approach: preserve the edges and corners even with superior performance of noise removing.

Lower left corner of figure 4 is the image restored by median filter with 5x5 window. Note that 5x5 window gives lowest mean square error as well as 3x3 window in the experiment of former section. Most of impulse noise are removed in this picture but it is much more blurred than the result of robust model based method. This blurring effect can be more easily observed in the images with many edges and corners than in the images with large areas with constant grey levels. Guy wire and details of the bridge frame is blurred and cannot be observed in these median filtered images. The regions with small intensity variations are replaced by constant grey level and the transitions between different regions are rather abrupt. This effect is typical in the median filter and it is because median filter fails in smoothing images. These effect can be observed in tower region of the bridge.

Lower right corner of figure 4 is the image restored by α-trimmed mean filter with 5x5 window and α=0.15. This α trimmed mean filter has lower level of mean square error in the experiment in the former section. But this method gives even more blurred image than median filter. Edges and corners of the image convey more information to human perception and because of this, the image restored by α-trimmed mean filter is worse than median filter in the visual comparison even it has smaller mean square error. For example, tower and guy wire in the bridge is very blurred. It is also not very successful in removing impulse noise and has many residual noise caused by impulse noise. These unremoved noise can be observed in almost all images.

VI. SUMMARY AND CONCLUSIONS

Robust estimation in causal autoregressive model is considered in this paper. When the outliers are present only in the recursive generation process, the estimator is obtained by minimizing an appropriate loss function. This estimator is consistent and asymptotically normal. The iterative estimation algorithm presented in this paper converges to robust M-estimator. For the estimation with noisy observation, an iterative estimation algorithm which use the data cleaning procedure is developed.

An image restoration method based on robust modelling approach is also presented in this paper. This method can restore images from the impulse noise contaminations as well as Gaussian noise contaminations, and has an advantage of preserving all the details of the original image. Mean square error of the images restored by robust model based method is smaller than images restored by any other conventional methods. Qualitative comparisons show the superior performance of the robust model based method.

References

[1] W. K. Pratt, *Digital Image Processing,* John Wiley & Sons, Inc., New York, 1978.

[2] P. J. Huber, *Robust Statistics,* John Wiley & Sons, Inc., New York, 1981.

[3] B. Kleiner, R. D. Martin, and D.J. Thomson, "Robust Estimation of Power Spectra," *Journal of Royal Society,* Series B, Vol. 41, No. 3, pp.313-351, 1979.

[4] S. A. Kassam and H. V. Poor, "Robust Techniques for Signal Processing," *Proceedings of IEEE,* Vol. 73, No. 3, pp.433-481, March 1985.

[5] Y. H. Lee and S. A. Kassam, "Generalized Median Filtering and Related Nonlinear Filtering Techniques," *IEEE Trans. on Acoustics, Speech and Signal Processing,* Vol. ASSP-33, No. 3, pp.672-683, June 1985.

[6] A. C. Bovick, T. S. Huang and D. C. Munson, Jr, "A Generalization of Median Filtering Using Linear Combinations of Order Statistics," *IEEE Trans. on Acoustics, Speech and Signal Processing,* Vol. ASSP-31, No. 6, pp.1342-1350, December 1983.

[7] R.L. Kashyap, "Analysis and Synthesis of Image Patterns by Spatial Interaction Models," in *Progress in Pattern Recognition,* (Edited by) L.N. Kanal and A. Rosenfeld, Vol. 1, North Holland, pp. 149-186, 1981.

[8] J.W. Tukey, *Exploratory Data Analysis,* Addison-Wesley Publ., Mass, 1971.

[9] R.E. Nasburg and R.L. Kashyap, "Robust Parameter Estimation in Dynamic Systems," *Proc. 1975 Conference on Information Sciences and Systems,* Baltimore, MD, 1975.

[10] Y. Yasuoka and R.M. Haralick, "Peak Noise Removal by a Facet Model," *Pattern Recognition,* Vol. 16, No.1, pp. 23-29, 1983.

[11] R. Chellappa and R.L. Kashyap, "Digital Image Restoration Using Spatial Interaction Models," *IEEE Trans. on Acoustics, Speech and Signal Processing,* Vol. ASSP-30, No. 3, pp.461-472, June 1982.

[12] Kundu, A., Mitra, S.K., and Vaidyanathan, P.P., "Application of Two-Dimensional Generalized Mean Filtering for Removal of Impulse Noises from Images," *IEEE Trans. on Acoustics, Speech and Signal Processing,* Vol. ASSP-32, No. 3, pp.600-609, June 1984.

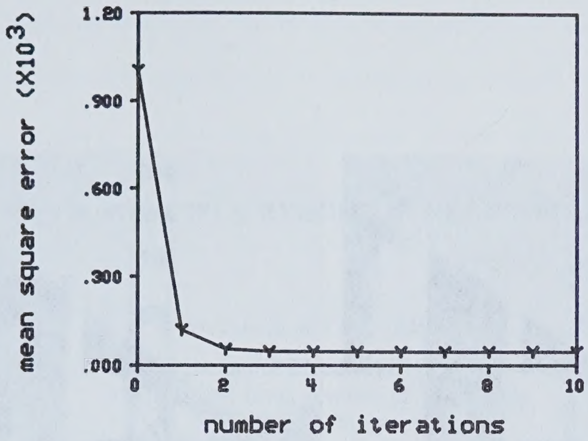


Figure 2. Convergence of mean square errors.

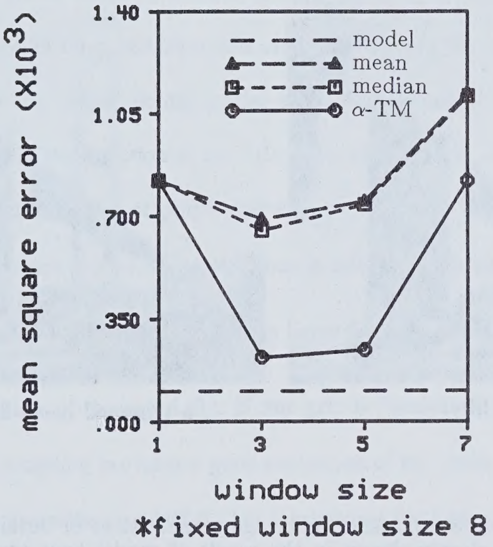
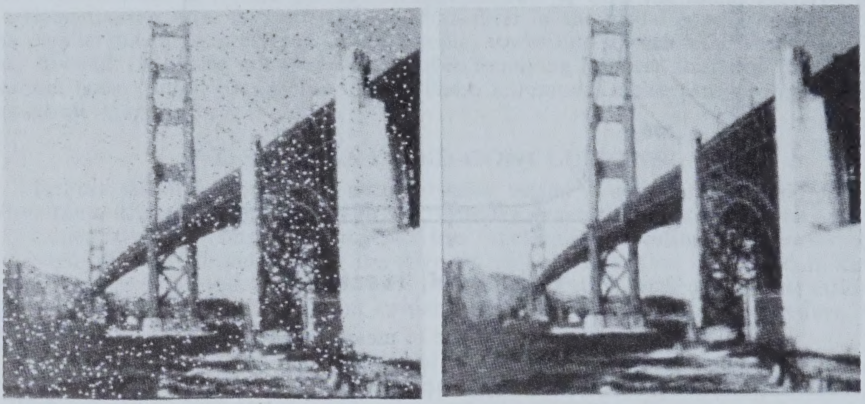
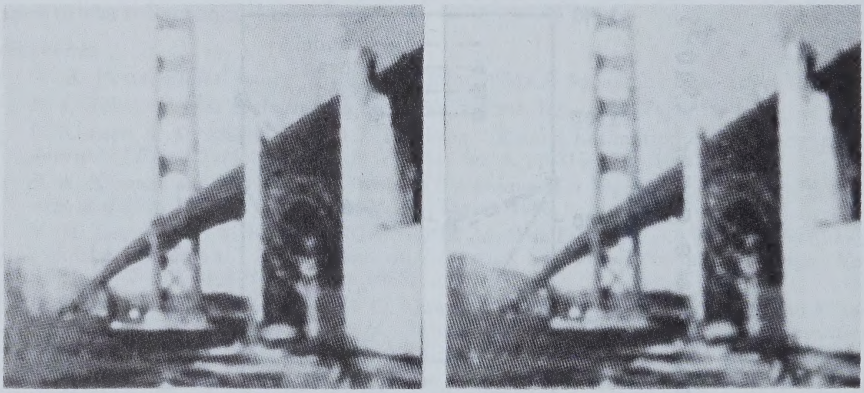


Figure 3. Mean square error comparison of different methods.



contaminated image

robust model approach



median filter

α-trimmed mean filter

Figure 4. Qualitative comparison for bridge scene. Most of details, such as guy wire, are clearly shown in the result of model base approach but are not clear in others.

ON A MULTIPLE REGISTRATION PROBLEM

A. Bruckstein and M. Lindenbaum

Department of Electrical Engineering
TECHNION, Israel Institute of Technology
Haifa, 32000, Israel

SUMMARY

Suppose a planar shape S, colored in black on a white background, is sampled along several parallel lines, resulting in a set of black segments on the sampling lines. Denote by $L(\theta)$ the set of points in the plane that belong to the K parallel lines $\{l_j(\theta), j=1,2,...,K\}$ in the direction θ, i.e. $L(\theta) = \{(x,y) | (x,y) \in l_j(\theta), j=1,2,...,K\}$. The sampling process yields the set of points defined by $M(S,\theta) = S \cap L(\theta)$. If the shape is sampled along several groups of parallel lines in different directions, say θ_i, $i=1,2,...,P$, then the set $\bigcup_{i=1}^{P} M(S,\theta_i)$ provides information equivalent to sampling S on a grid determined by the union of the sets $L(\theta_i)$. If this grid is "dense" in the region where the object lies, this sampling provides a good evaluation of the shape S. Suppose, however that we are not given the sets $M(S,\theta_i)$ but rather some randomly translated versions of these, i.e. we only have access to $M^*(S,\theta_i) = \{M(S,\theta_i)+V_i\} = \{(x,y)+V_i | (x,y) \in M(S,\theta_i)\}$, where $\{V_i\}$ is a set of unknown vectors. Such a situation arises if the object makes random translatory moves between the samplings or if we do not know the relative

positions of the sets of parallel lines along which the object was sampled. The question that now arises is whether there exists a way to integrate the samples $M^*(S,\theta_i)$ so as to get good evaluations of the shape that generated them. Of course, ideally we would like to somehow determine the displacements V_i and then take the union of $\{M^*(S,\theta_i) - V_i\}$.

Let us place the sets $M^*(S,\theta_i)$ assuming that they were displaced by the vectors U_i. There is a simple way to check whether the assumed set of displacement vectors $\{U_i\}$ could be the true set $\{V_i\}$. Consider the grid generated by the union of lines $\bigcup_{i=1}^{P}\{L(\theta_i) - U_i\}$ and define the set of intersection points in this grid to be $C_G(\{U_i\})$. Let $C_B(\{U_i\})$ be the set of (black segment) intersection points in $\bigcup_{i=1}^{P}\{M^*(S,\theta_i) - U_i\}$. The vectors U_i have been incorrectly chosen if

$$\left[\bigcup_{i=1}^{P}\{M^*(S,\theta_i)-U_i\}\right]\cap C_G(\{U_i\})\neq C_B(\{U_i\})$$

since this implies the existence of points in the plane that, according to one of the assumed samples, say $\{M^*(S,\theta_i) - U_i\}$, belong to S, whereas according to some other sample they belong to the exterior of S. Therefore we know how to test a set of placement vectors $\{U_i\}$ for "correctness". If such a set of placement vectors is not ruled out as incorrect by the preceding test, then there exists a shape S that could have produced the measurements $\bigcup_{i=1}^{P}\{M^*(S,\theta_i)-U_i\}$ and we evaluate that shape by looking at the set of (suitably placed) measurements.

The problem that remains to be solved is the following: Given the measurements

$M^*(S,\theta_i)$, determine sets of vectors that yield legal placements of the measurements in the plane, according to the above discussed rule. We can choose $U_1 = 0$, since only the relative positions of the measurements is important. It is also clear that, for the overall placement to be correct, it has to yield correct relative placements of each pair of measurements. So $U_j - U_i$ has to belong to the range of correct placements of the j'th measurement w.r.t. the i'th one. We shall denote these "binary" ranges of correct placements by $R\{j|i\}$ (obviously $R\{j|i\} = -R\{i|j\}$). Therefore we have the following procedure for determining a "correct" overall placement $\{U_i\}$.

A. For each pair of measurements $\{M^*(S,\theta_i),M^*(S,\theta_j)\}$ determine the legal placement range as the set of translation vectors that yield correct binary placements according to the "intersection" criterion.

B. Determine a set of vectors $U_1 = 0, U_2, \ldots, U_P$, so as to have $U_j - U_i \in R\{j|i\}$ for all pairs i,j. Determining the placements $\{U_i\}$ according to this rule is a *multiple registration problem.*

In our case it is easy to see that the binary legal placement ranges $R\{j|i\}$ are collections of parallelogram-shaped areas. In general, if for all i and j, $R\{j|i\}$ is the interior of a convex polygon then the registration problem, i.e. the determination of a feasible placement, can be solved via linear programming. For placement ranges that are collections of convex polygonal shapes we have to solve a linear programming problem for each combination of simple convex shapes, to determine feasible solutions. If the basic ranges are discrete sets of vectors, the determination of a feasible solution can only be done by exhaustive search, the problem being NP-complete (reduction to partitioning).

We note that it would be interesting to find an algorithm that, given $R\{j|i\}$, determines the regions $\tilde{R}_i$ for $i=2,3,..,P$, having the property that for *any* $U_i^* \in \tilde{R}_i$ there exists a feasible solution of the registration problem having $U_i = U_i^*$. The regions $\tilde{R}_i$ measure the total uncertainity about V_i, given the randomly translated sampling data. Also, it would be interesting to find efficient solutions for the registration problem in other particular cases of placement ranges $R\{j|i\}$, like general convex regions (circles, ellipses, etc).

The issues discussed in this note arose in our investigation on determining the shape of moving objects from sparse samplings of edge-crossing measurements.

REFERENCE

M. Lindenbaum and A. Bruckstein, "Determining Object Shape from Local Velocity Measurements", EE. Report 599, Technion, IIT, Haifa, June 1986.

FIGURES

FIGURE 1: A planar shape and its samples in 4 directions.

FIGURE 2: Ranges of correct "binary" placements $R\{j|i\}$.

FIGURE 3: A legal and an illegal placement for samples 1 and 2.

FIGURE 4: Complete legal registrations produced by linear programming.

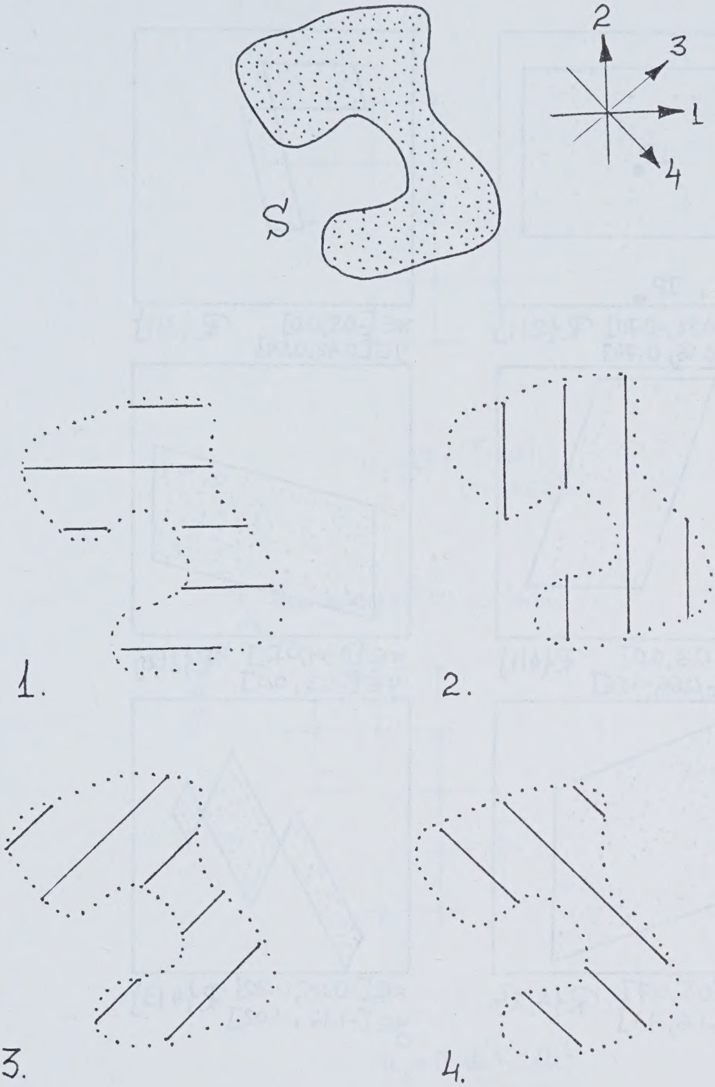


FIGURE 1: A planar shape and its samples in 4 directions.

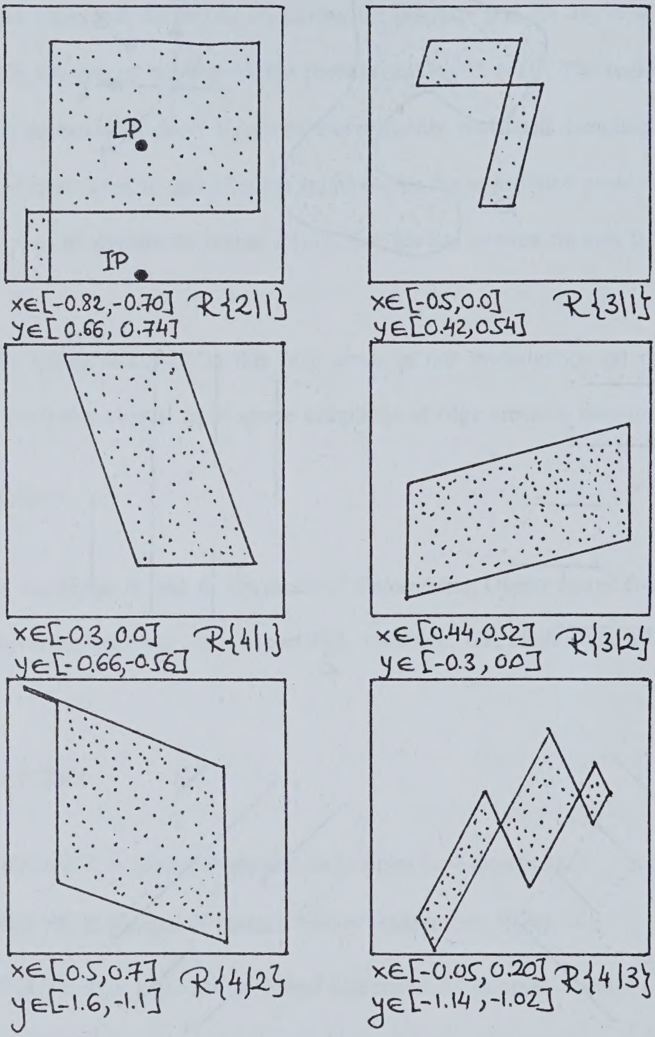
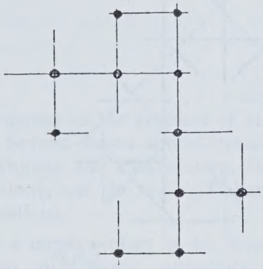
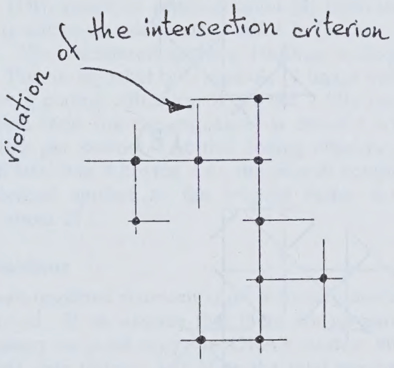

LP
IP
x∈[-0.82,-0.70] R{2|1}
y∈[0.66, 0.74]
x∈[-0.5,0.0] R{3|1}
y∈[0.42,0.54]
x∈[-0.3,0.0] R{4|1}
y∈[-0.66,-0.56]
x∈[0.44,0.52] R{3|2}
y∈[-0.3 , 0.0]
x∈[0.5,0.7] R{4|2}
y∈[-1.6,-1.1]
x∈[-0.05,0.20] R{4|3}
y∈[-1.14 ,-1.02]

FIGURE 2: Ranges of correct "binary" placements $R\{j|i\}$.



$u_2 = LP \in R\{2|1\}$
(FIGURE 2)



violation of the intersection criterion

$u_2 = LP \notin R\{2|1\}$
(FIGURE 2)

FIGURE 3: A legal and an illegal placement for samples 1 and 2.

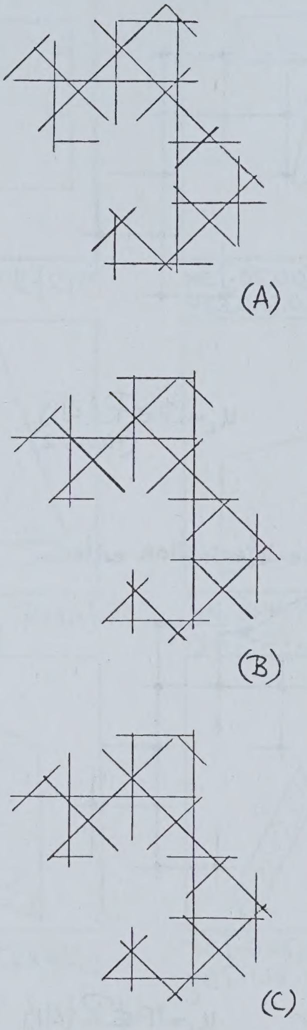


FIGURE 4: Complete legal registrations produced by linear programming.

Encoding Biomedical Images with Quadtrees

Daniel R. Fuhrmann*
Michael A. Brown**
Institute for Biomedical Computing
*Dept. of Electrical Engineering
**Dept. of Computer Science
Washington University
St. Louis, Mo.

1. Introduction

We describe work in progress on the problem of encoding multicolored segmented cell images using quadtrees [1]. Several issues are addressed, namely the efficiency of various coding techniques for data storage and transmission, the internal storage requirements of different quadtree representations, and the trade-offs between time and space complexity of various image processing operations.

This research is part of a larger project in the automation of a biological experimental technique known as electron microscopic autoradiography (EMA) [1]. This technique involves the analysis of electron microscope cell images at very high magnification. The cell image is to be segmented into a number of regions of biological interest, e.g., cytoplasm, nucleus, mitochondria, etc. The analysis of the segmented image requires a numerical 2-dimensional integration of a probability density function over the various irregularly shaped regions in the image. A typical image for this project has 200k pixels (bytes) and the corresponding quadtree has 30k nodes.

2. Coding for Data Storage and Transmission

The depth-first (DF) quadtree representation [4] provides a natural starting point for compression of the quadtree-encoded image since it forms a sequence of symbols from a small finite alphabet. We considered applying Huffman coding and the Lempel-Ziv method [5] to this sequence. For an alphabet of 8 symbols (7 image colors plus GRAY), we find that both methods achieve a coding efficiency of around 2 bits per symbol for our cell images. Hence the compression ratio (on the sequence) is about 1.5:1, assuming an uncompressed representation of 3 bits per symbol. At this coding efficiency one 200kbyte image can be stored externally with less than 8 kbytes, i.e., the overall compression ratio is about 25:1. In contrast, the LZ method applied to the original raster scan representation achieves a compression ratio of about 21:1.

3. Internal Representations

The pointer-based quadtree representation is usually described as being very inefficient; this is not necessarily so. If we assume that there are no parent pointers in our tree structure, then it is necessary to store only the GRAY nodes: all leaves of one color can be represented by a single data object. Let N be the total number of nodes in the tree. If we assume 4-byte pointers, then each GRAY node requires 20 bytes (4 pointers plus 1 color plus 3 wasted bytes for alignment). The GRAY nodes comprise 1/4 of the tree, hence the total space required is $5N$ bytes. For 2-byte pointers (sufficient to encode a tree with 256k nodes), this number becomes $2.5N$ bytes.

The linear quadtree [5] requires a list of location codes for all the leaves of the tree. This location code requires $2M$ bits, where M is the depth of the tree. For $M = 9$ or 10 (10 corresponds to a 1k × 1k image) we require 3 bytes per leaf, with the color of the leaf stored in a 4th byte. The leaves comprise 3/4 of the tree, hence the required storage is $3N$ bytes.

The DF representation is the most efficient for internal storage. If the byte is the smallest data type with which we wish to work, then this representation requires N bytes.

4. Image Processing Operations

For the EMA project, we find that the operations to be performed on the quadtree-encoded images fall into three categories: 1) those that visit every node of the tree in a preorder traversal, 2) those that visit every node and some of its neighbors, and 3) those that modify the tree.

Operations in category (1) include image display, numerical 2-dimensional integration over irregularly shaped regions, and image encoding as described in Section 3. For these operations, the DF representation is a natural choice since it contains all the necessary information in the order in which it is processed.

Operations from category (2) are considered by Samet in [6]; an example is connected component labelling, which we are using as a contour filling algorithm. This algorithm requires that arbitrary sons (not just NW) of any GRAY node be accessed at any time. The pointer-based representation is simplest for this type of operation, although the DF representation could be used at the expense of extra scans of the data [7].

For modifying or building the tree, as is done during interactive data acquisition, the pointer-based structure is by far the simplest to manipulate. Any changes to the linear quadtree or DF representation would require the shifting of large sections of the data in memory. In addition, operations such as image intersection which require searches along arbitrary paths in the tree would become most time-consuming.

5. Summary

The quadtree is an effective data structure for representing segmented images in automated EMA experiments. For data storage and transmission, the DF representation, further encoded by standard data compression techniques, has led to compression ratios of better than 25:1. For image acquisition, modification, and for other processing algorithms, the pointer-based representation is simple to implement and is a reasonable alternative in terms of internal storage.

References

1. H. Samet, "The Quadtree and Related Hierarchical Data Structures", *ACM Computing Surveys*, vol. 16, June 1984, pp. 187-260.
2. M. Miller et. al., "Maximum Likelihood Estimation Applied to Electron Microscopic Autoradiography", *J. Electron Microscopy Technique* 2:611-636 (1985).
3. I. Gargantini, "An Effective Way to Represent Quadtrees", *Comm. ACM*, vol. 25, Dec. 1982, pp. 905-910.
4. E. Kawaguchi et. al., "Depth-First Picture Expression Viewed from Digital Picture Processing", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-5, July 1983.
5. J. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression", *IEEE Trans. Information Theory* , vol. IT-23, May 1977.
6. H. Samet, "A Top-Down Quadtree Traversal Algorithm", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-7, Jan. 1985.
7. D. Fuhrmann, "Quadtree Traversal Algorithms for Pointer-Based and Depth-First Representations", in review, *IEEE Trans. Pattern Analysis and Machine Intelligence*.

ADAPTIVE SEGMENTATION OF NOISY IMAGES: AN EM ALGORITHM APPROACH

Haluk Derin and Sridhar Lakshmanan
Department of Electrical and Computer Engineering
University of Massachusetts
Amherst, Massachusetts 01003

Abstract

Given a noisy image modeled as a discrete-valued Markov random field (MRF) corrupted by additive independent Gaussian noise, the adaptive segmentation scheme presented here estimates the model parameters and segments the noisy image iteratively. The iterations are in the spirit of the EM Algorithm (EM-A) and the maximizations are carried out via simulated annealing (SA).

Image Model and Problem Statement

Let all random fields be defined on $N_1 x N_2$ rectangular lattice L, the observed image y (realization from r.f. Y) be the sum of underlying scene x (realization from r.f. X) and noise w (realization from r.f. W) and W consist of i.i.d. $N(0,\sigma^2)$ distributed r.v.'s. Let the process X be a MRF with a neighborhood system $\eta=\{\eta_{ij}:(i,j)\varepsilon L\}$ and each X_{ij} take one of M values from $\{r_1,r_2,\ldots,r_M\}$. Furthermore, let $P(X=x)>0$ for all x, so that the joint distribution of X is a Gibbs distribution (GD), i.e., $P(X=x) = \frac{1}{Z}\exp\{-\sum_c V_c(x)\}$, where Z is the normalizing constant, c is a clique (single pixel or set of pixels that are neighbors of each other) of (L,η) and $V_c(x)$ is the potential (function) associated with the clique c. The special class of GD considered here, which we call the multi-level logistic (MLL) distribution, has:

$$V_{c_k}(x) = \begin{cases} -\beta_k & \text{if all } x_{ij} \text{ in } c_k \text{ are equal} \\ \beta_k & \text{otherwise} \end{cases}$$

and for single pixel cliques

$$V_c(x) = \alpha_\ell \quad \text{if } x_{ij} = r_\ell \quad \text{for } (i,j)=c$$

The α's and β's together constitute the model parameters, denoted by Θ.

To express the parameter dependence, the joint distribution is denoted as $P(X|\Theta)$. If Θ were known, then we would seek a MAP segmentation, that is, $x^* = \arg\max_x P(X|Y,\Theta)$ as in [1-3]. But since Θ is unknown, we devise an iterative scheme which simultaneously estimates the parameters and performs a MAP segmentation. In other words, it puts out a sequence of estimates $(\hat{x}^k,\hat{\Theta}^k)$ that converges to (x^*,Θ^*) which maximizes $P(X,Y|\Theta)$.

Adaptive Segmentation Algorithm

The EM-A [4] is an iterative procedure consisting of the two steps:

E-Step: $\hat{x}^k = E\{X|Y=y,\hat{\Theta}^{k-1}\}$

M-Step: $\hat{\Theta}^k = \arg\max_\Theta P(X=\hat{x}^k|\Theta)$

and the sequence $(\hat{x}^k,\hat{\Theta}^k)$ converges to $\arg\max_\theta P(Y=y|\Theta)$ for Θ and $E\{X|Y=y,\Theta\}$ for x with Θ being the MLE of Θ from $P(Y=y|\Theta)$.

For the problem at hand, neither step of the EM-A is directly implementable and hence some modifications are suggested. We want (x^*,Θ^*) that maximizes $P(X,Y|\Theta)$. So, assuming everything else is fixed, we update first x_{ij}'s, then components of Θ, then x_{ij}'s again, and so on. For computational reasons, however, we do several updates of x_{ij}'s and then several updates of Θ, and so on. In this scheme the updating of x and Θ can be viewed as E and M steps of the EM-A, respectively; or as an attempt to obtain the MAP segmentation by SA, where the SA iterations are periodically interrupted for the updating of the parameters. The updating of the parameters, on the other hand, are done by a second level of SA, implementing the coding method [5]. The updating of the x is done by SA as in [2].

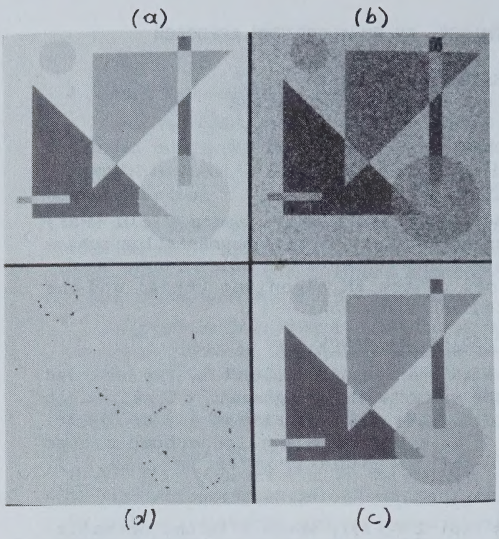


Fig.1 Segmentation of a 4-level geometric image: (a) original image; (b) noisy image (SNR=2); (c) segmentation; (d) error plane.

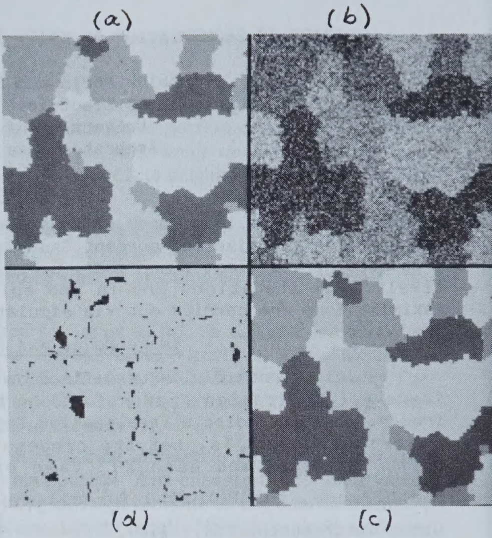


Fig.2 Segmentation of a 4-level Gibbsian image: (a) original image; (b) noisy image (SNR=1); (c) segmentation; (d) error plane.

This particular implementation of the coding method for parameter estimates turns out to be a very reliable scheme, much faster (increases linearly in complexity with the number of parameters) than the conventional methods which deal with the solution of nonlinear ML equations. A similar iterative algorithm estimating the parameters and segmenting the noisy image has been presented in [6], but was not implemented. Hence the applicability and the performance of that algorithm could not be attested.

Results and Conclusions

Due to space limitations only a few results are presented. Figs. 1 and 2 depict two applications of the adaptive segmentation algorithm where the estimation of the GD parameters and the segmentation are carried out iteratively and simultaneously. To the best of our knowledge, this is the first algorithm which implements parameter estimation and segmentation simultaneously. Considering the difficulty of the problem, it constitutes a significant achievement towards a fully data-driven segmentation algorithm. A detailed account of the algorithm and its application to several examples will be presented in a forthcoming paper. The convergence properties of this algorithm are of considerable theoretical interest and are being investigated.

References

[1] Elliott, H., Derin, H., Cristi, R. and Geman, D., "Application of the Gibbs Distribution To Image Segmentation", ICASSP'84, San Diego, CA, March 1984.

[2] Geman, S. and Geman, D., "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images", IEEE PAMI, pp.721-741, Nov. 1984.

[2] Derin, H. and Elliott, H., "Modelling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields", to appear in IEEE PAMI, Nov. 1986.

[4] Dempster, A.P., Laird, N.M. and Rubin, D.B., "Maximum Likelihood from Incomplete Data via the EM Algorithm," J.R.Stat.Soc., Ser.B, V.39, pp.1-38, 1977.

[5] Besag, J. E., "Spatial Interaction and the Statistical Analysis of Lattice Systems", J.R.Stat.Soc., Ser.B, V.36, pp.192-236, 1974.

[6] Geman, D., "Bayesian Image Analysis by Adaptive Annealing", Dig. IGARSS'85, Univ. of Mass., Amherst, MA, Oct. 1985.

AN APPLICATION OF TENSOR THEORY TO 3-D SHAPE ANALYSIS†

NIRWAN ANSARI and EDWARD J. DELP
Purdue University
School of Electrical Engineering
West Lafayette, IN 47907
(317) 494-1740

Abstract

In this paper, we present two new approaches to address the shape recognition problem. The first concept is an extension of 2-D landmark-based shape analysis to the 3-D case. Secondly, we extend the work of Cyganski and Orr to rigid non-planar objects. Both approaches will be mathematically formulated, however no experimental results will be shown in this paper.

I. 3-D Landmark-based Shape Analysis

Objects can be represented by landmarks or tokens in 3-D space. Landmarks or tokens are points, usually defined along the object surface, that have important shape attributes. These attributes are chosen a priori and are problem dependent. Example of landmarks are corners and points of high curvature. We shall assume throughout this paper that the landmark locations have been detected. We shall address the problem of processing the landmarks to identify the object. To identify an observed (unknown) object from a library of objects, we find the necessary transformation that maps landmarks of each library object to the corresponding landmarks of the observed object. It is difficult, even if it exists, to determine this transformation. If the transformation can be found, then shape features based on the transformation can be extracted and used for object identification. We shall address this problem similar to the way we addressed the problem of 2-D shape analysis [AnD85]. For 2-D shape analysis the object was partitioned into triangles and the mapping used was triangular deformation obtained through an affine transformation. The shape features in the 2-D case were based on the dilatations obtained from the transformation. In 3-D we partition the object into tetrahedra and use the tetrahedral deformation, i.e. a transformation from one tetrahedron to another.

I.1. Quasiconformal mapping

Tetrahedral deformation can be determined by a 3-D affine transformation. The 3-D affine transformation is a class of quasiconformal mappings in 3-D which are defined [Vai61] as follows:
A diffeomorphism, $f:\Omega\rightarrow\Omega$, ($\Omega, \Omega \subset \mathbf{R}^3$) is called K-quasiconformal if

$$\sup_{x\epsilon\Omega} D(x) \leq K < \infty \qquad \text{for some constant K,}$$

$$\text{where} \quad D(x) = \max(\bar{D}(x),\underline{D}(x)) \ ,$$

$$\bar{D}(x) = \frac{(\max_{|e|=1} |f'(x)e|)^3}{|J(x)|} \ , \qquad \underline{D}(x) = \frac{|J(x)|}{(\min_{|e|=1} |f'(x)e|)^3} \ ,$$

$J(x)$ and f' are the Jacobian and the derivative of f respectively. Note that $\max_{|e|=1} |f'(x)e|$ and $\min_{|e|=1} |f'(x)e|$ correspond to the largest and the smallest eigenvalue of the Jacobian matrix of f. $\bar{D}(x)$,$\underline{D}(x)$, and $D(x)$ are known as the outer, inner and maximal dilatations of f, and are related to the eigenvalues of the Jacobian matrix of f by the above equations. Infinitesimally f maps a ball to an ellipsoid. These dilatations at each $x\epsilon\Omega$ are various combinations of the ratios of the semi-axes of the 3-D ellipsoid. In general, it is difficult (if it exists) to find a diffeomorphic map of a library object to an observed object. The affine transformation from one tetrahedron to another is however uniquely determined. Since the 3-D affine transformation is a class of quasiconformal mappings, we can use the above definitions to obtain the dilatations. The maximal dilatation derived from the affine transformation can be used as a distortion measure between the two tetrahedra (they are similar if the maximal dilatation is 1). Using an approach similar to that of partitioning a 2-D object into triangles, we partition a 3-D object into tetrahedra. We then evaluate the maximal dilatation and the scale change (magnitude of the Jacobian) of each tetrahedral deformation between a library object and the observed object. If the two objects are similar, for each tetrahedral deformation between the two objects, the maximal dilatation is one and the scale change remains constant. Hence, the standard deviations of the maximal dilatation and the scale change associated with all the tetrahedral deformations between the two objects can be used to determine how similar the two objects are. The smaller the deviations are, the more similar the two objects are.

I.2. Strain tensors

The above concept can similarly be formulated by tensor analysis. We assume that the observed object is deformed from a library object. Denote Ω (with respect to reference frame X, x^i, i=1,2,3) as the region occupied by the library object, and Ω (with respect to reference frame X,

---

† This research was partially supported by the Air Force Office of Scientific Research under Grant F49620-86-K-006 and the National Institutes of Health under Grant No. 1RO1 HL29716-01A1.

$\bar{x}^i$, i=1,2,3) as that of the observed object. We assume that the deformation from Ω to $\bar{\Omega}$ is homeomorphic. Assuming both X and X are orthogonal cartesian, the square of the arc length of the library and observed object using the general summation rule, are respectively

$ds^2 = dx^i dx^i$ and $d\bar{s}^2 = d\bar{x}^i d\bar{x}^i$ i=1,2,3.

Thus, $d\bar{s}^2 - \quad = 2\mu_{ij} dx^i dx^j$ i,j=1,2,3. and $\mu_{ij} = \frac{1}{2}\left(\frac{\partial \bar{x}^k}{\partial x^i}\frac{\partial \bar{x}^k}{\partial x^j} - \delta_{ij}\right)$.

The tensor μ_{ij}, which is symmetric, is called the strain tensor [Sok51]. The eigenvalues of μ_{ij} are called the principal strains, and the corresponding eigenvectors are orthogonal. The directions corresponding to the principal strains are called the principal directions (axes) of the strain tensor. At any given point, it is a mapping from a ball to an ellipsoid. The ratio of each semi-axis of the ellipsoid to the diameter of the ball equals $\sqrt{1+2\lambda_i}$, i=1,2,3, where λ_i's are the eigenvalues of μ_{ij}. Again, the deformation from a library object to the observed object may not be homeomorphic. It is difficult to determine the deformation function even if it is homeomorphic. The tetrahedral deformation used for the above landmark-based problem can however be uniquely determined. The various dilatations (inner, outer, maximal) of the tetrahedral deformation can also be easily computed by means of the eigenvalues of the strain tensor associated with the tetrahedral deformation. The strain tensor analysis which provides an intuitive analysis of deformation thus can also be used to solve the landmark-based problem.

II. Surface tensors

We shall develop another concept for object identification dealing with objects that are characterized by surface patches rather than sets of landmarks. Object identification becomes a problem of identifying surface patches. Let Ω with coordinate reference frame X be a surface occupied by a library object, and $\bar{\Omega}$ with coordinate reference frame X be a surface of an observed object. This notation is consistent with the strain tensor notation presented above, the difference being that Ω is a surface instead of a set of landmarks. We further assume that Ω and $\bar{\Omega}$ are related by an affine transformation, i.e. $\bar{x}^j = a_i^j(x^i - c^i) + \bar{c}^j$, where c^i and $\bar{c}^i$ are the centers of gravity of Ω and $\bar{\Omega}$. Here, we are interested in finding the a_i^j, the coefficients of the affine transformation. We can thus transform the library object (surface) with respect to the observed object (surface) by the affine transformation. The similarity between the library object (surface) and the observed object (surface) is determined by how the transformed library object (surface) is correlated with the observed object (surface). We describe how to determine the affine coefficients below. We form a surface (moment) tensor on Ω and $\bar{\Omega}$ as follows:

$$S^{ij\cdots} = \int_{\Omega}(x^i - c^i)(x^j - c^j)\cdots dx^1 dx^2 dx^3 = \int (x^i - c^i)(x^j - c^j)\cdots I_{\Omega}(x^1,x^2,x^3)dx^1 dx^2 dx^3$$

$$S^{ij\cdots} = \int_{\bar{\Omega}}(\bar{x}^i - \bar{c}^i)(\bar{x}^j - \bar{c}^j)\cdots d\bar{x}^1 d\bar{x}^2 d\bar{x}^3 = \int (\bar{x}^i - \bar{c}^i)(\bar{x}^j - \bar{c}^j)\cdots I_{\bar{\Omega}}(\bar{x}^1,\bar{x}^2,\bar{x}^3)d\bar{x}^1 d\bar{x}^2 d\bar{x}^3$$

$$= \int_{\Omega} a_k^i(x^k - c^k)a_l^j(x^l - c^l)\cdots |J| dx^1 dx^2 dx^3 = a_k^i a_l^j \cdots |J| S^{ij\cdots}$$

$$\text{where } I_{\Omega}(x^1,x^2,x^3) = \begin{cases} 1 & \text{if } (x^1,x^2,x^3) \in \Omega \\ 0 & \text{otherwise} \end{cases}, \quad I_{\bar{\Omega}}(x^1,x^2,x^3) = \begin{cases} 1 & \text{if } (\bar{x}^1,\bar{x}^2,\bar{x}^3) \in \bar{\Omega} \\ 0 & \text{otherwise} \end{cases}$$

$$a_i^j = \frac{\partial \bar{x}^j}{\partial x^i} \qquad |J| = \left|\det\left(\frac{\partial \bar{x}^j}{\partial x^i}\right)\right|$$

Thus, $S^{ij\cdots}$ is a relative tensor of weight -1. Following the above steps we form various orders of surface tensors for both Ω and $\bar{\Omega}$, we then contract them to unit rank absolute tensors similar to [CyO85]. Three pairs of such unit rank absolute tensors of Ω and $\bar{\Omega}$, say, u^i, $\bar{u}^i$, v^i, $\bar{v}^i$, w^i, $\bar{w}^i$, are required to form a system of linear equations:

$\bar{u}^i = a_j^i u^j$, $\bar{v}^i = a_j^i v^j$, $\bar{w}^i = a_j^i w^j$

such that we can obtain a_j^i.

REFERENCES

[AnD85] N. Ansari and E. J. Delp, "An approach to landmark-based shape recognition," *Proc. 23rd Allerton Conference on Communication, Control and Computing,* Monticello, IL., pp.292-300, October 2-4, 1985.

[CyO85] D. Cyganski and J. A. Orr, "Applications of tensor theory to object recognition and orientation determination," *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol PAMI-7, pp.662-673, Nov. 1985.

[Sok51] I. S. Sokolnikoff, *Tensor Analysis,* Wiley & Sons, 1951.

[Vai61] J. Vaisala "On quasiconformal mappings in space" *Annales Academic Scientiarum Fennice,* A.I. 298, 1961.

REAL-TIME PATTERN RECOGNITION WITH DISCRETE FOURIER PREPROCESSING TRANSFORMS

O. K. Ersoy, Douglas Y. Kim

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

INTRODUCTION

In this paper, we introduce new techniques of real-time transform-based pattern recognition which drastically decrease the time of computation and cost of implementation and simultaneously increase the quality of recognition in the presence of noise, as compared to "conventional" transform-based techniques.

The new techniques will be collectively referred to as the discrete Fourier preprocessing transforms (DFPT) method. Both software and hardware issues for the real time implementation of the recognition tasks are handled with the DFPT method.

In this introductory article, the DFPT method is presented as a modification of the Fourier descriptor (FD) method [2]. In the FD method, one takes the discrete Fourier transform (DFT) of the contour of a closed boundary which is modeled in terms of a chain code [2]. The FD method is invariant to object translation and rotation. It can also be made approximately invariant to object scaling by a proper normalization procedure. However, the FD techniques are computationally intensive, and costly to implement in real time.

THE DFPT METHOD FOR OBJECT RECOGNITION

The DFPT method is an outgrowth of our research in the area of two-stage algorithms to represent discrete trigonometric transforms such as discrete Fourier (DFT) and real discrete Fourier (RDFT) transforms [1]. DFPT will be mainly utilized for the purpose of feature extraction followed by a classification algorithm.

Discrete Fourier preprocessing transforms (DFPT's) can be defined as the modification of the discrete Fourier (DFT) or real discrete Fourier (RDFT) transform such that the kernel functions $\sin 2\pi n/N$ and $\cos 2\pi n/N$ are replaced by $z(n/N)$ and $z(n/N+1/4)$, respectively, where $z(\cdot)$ is a function satisfying certain constraints. $z(n/N)$ is isomorphic to $\sin 2\pi n/N$, and $z(n/N+1/4)$ is isomorphic to $\cos 2\pi n/N$.

$z(\cdot)$ is chosen such that the DFPT matrix contains elements which require only add and/or shift operations. There are also data structures for the fast computation of DFPT. Since there are no multiplications, VLSI or electro-optical implementations of DFPT are especially attractive.

We developed normalization procedures such that the method with DFPT in replacement of DFT is invariant to rotation, translation and scaling of the object. It is also invariant to the starting point of the transform on the object contour.

EXPERIMENTAL RESULTS

Our computer simulations with the method of Fourier descriptors for shape recognition indicate that DFPT performs considerably better than DFT in image recognition, especially when the image is embedded in noise. In our experiments, we chose $z(\cdot)$ to be equal to the rectangular wave function [1]. The resulting method is referred to as the discrete rectangular wave transform (DRWT) method.

The first experiment was to classify the measured contour as one of 6 different aircraft. The total number of features available was 512. The 32 largest features were chosen in the transform space and the Euclidean distances for classification were calculated according to both methods. The experiment was repeated 6 times. In each case, the test object was changed, thus covering all 6 possible objects. The DRWT method proved to be much better in terms of differentiation between the test object and the library objects. Next we investigated the performance of the two methods in the presence of noise. The x and y coordinates of the contour of each aircraft were corrupted independently by Gaussian noise corresponding to 30dB, 20dB, 10dB and 3dB SNR, respectively.

The table below shows the percentage of correct recognition with the two methods.

SNR	Fourier Descriptors	DRWT Method
3dB	50.0%	83.3%
10dB	66.7%	100.%
20dB	66.7%	100.%
30dB	83.37%	100.%
∞	100.%	100.%

It is clear from the table above that the DRWT method achieves considerably better accuracy than the Fourier descriptor method in the presence of noise. We will try to give a brief explanation of these results. In both methods, 32 largest coefficients are chosen as the features. In the method of the Fourier descriptors, these are the low frequency components, meaning that high frequency information is discarded. In the DRWT method, most of the largest coefficients are also the low order coefficients in the DRWT domain, but these are also combinations of Fourier spectral components such that they contain both low and high frequency Fourier spectral information. As such, they represent the true shape more faithfully.

Other experiments performed in the presence of object rotation, translation and scaling also showed that the DFPT method is more robust than the FD method when proper normalization procedures are used.

Thus, the requirements of simplicity, very high speed, low cost, and high accuracy of recognition in complicated scenes can be simultaneously satisfied with the DFPT method.

REFERENCES

[1] O.K. Ersoy, "Representation of real discrete Fourier transform in terms of a new set of function based upon Mobius inversion", Bit, v.25, pp. 541-556, 1985.

[2] T.P. Wallace and P.A. Wintz, "An efficient three-dimensional aircraft recognition algorithm using normalized Fourier descriptors", *Comput. Graphics Image Processing,* vol. 13, pp. 96-126, 1980.

FAIR PROOFS OF DISTRIBUTED PROGRAMS THAT ARE SAFE FROM CONSPIRACY

R.K.Shyamasundar
Department of Computer Science,
Pennsylvania State University,
University Park, PA 16801,
and
Computer Science Group,
Tata Institute of Fundamental Research,
Homi Bhabha Road ,
Bombay 400 005, India

ABSTRACT

In this paper, we provide a structured proof method for proving the livelock- freedom of distributed programs in spite of conspiracy. Our methodology uses the proof principles of the CSP programs and the proof of fair termination of CSP programs under serialized semantics. The method provides a two- level approach for proving the strong fairness of distributed programs in the presence of conspiracy.

1. INTRODUCTION

The *fairness problem* accounts for the most important respect in which the formal description of programming languages involving parallelism can be expected to be inadequate. The interest in fairness stems from the study of parallel programs where one wishes to express the fact that every component program will eventually finish the execution of each atomic instruction and start the next one. In the literature , the problem of proof rules for fair termination of programs has received considerable interest such as [4] , [7,8,9,10,11,12] . Though , the interest in *fairness* stems from concurrent programs , many of the results are formulated in terms of nondeterministic programs mainly because of the elegant formalisms of *nondeterminism* , such as Dijkstra's guarded commands and also because of the several elegant transformations from parallel to nondeterministic programs. Various notions of fairness have been formulated in [9] , [10] , and [12]. [7] has shifted the emphasis of fair-termination of nondeterministic programs to the context of communicating sequential processes by defining the notions of *serialized interleaving semantics* and *overlapping interleaving semantics.* Though various hierarchies of fairness have been defined , the methods do not cater to the correctness of general CSP programs using the interleaving model of semantics and further, the fairness issues in the presence of *conspiracies* is ignored. Intuitively , *conspiracy* among processes may be said to occur when a collection of processes act together in such a way as to prevent another process from making a particular move or action essential for achieving its goal. That is, conspiracy may prevent a process in progressing towards its goal. The possibility of conspiracy is one of the essential differences of a truly concurrent overlapping mode of execution from a merely interleaved serialized model. It may be noted that conspiracy can only occur in the truly concurrent model (i.e. overlapping) model as many processes could be executing their respective actions at a time (unlike the serial model where at most one handshaking communication can be taking place at any instant of time) and also, it must be noted that the notion of conspiracy is important particularly for *nonterminating* programs.

Back and Kurki-Suonio[5] explore the notion of conspiracy in the context of serializability of distributed systems for comparing serial and concurrent models under a temporal logic framework and Kuiper and de Roever[12] have defined various fairness assumptions depending on different implementations of the guarded command construction as well as on synchronized communication (note that both are features of CSP).

In this paper, we provide a structured proof method for proving the livelock- freedom of distributed programs in spite of conspiracy. Our methodology uses the proof principles of the CSP programs and the proof of fair termination of CSP programs under serialized semantics. The method provides a two- level approach for proving the strong fairness of distributed programs in the presence of conspiracy. We keep the discussion informal for lack of space.

2. FAIR PROOFS OF CSP PROGRAMS

One of the natural approaches for proving the fair-termination of programs is that of *helpful directions* as discussed uniformly in [6]. In the sequel, we essentially use the notation from [7,8,9].

Helpful Directions: Let us suppose that we are given a program S as an iteration of guarded transitions namely,

$$S :: *[\ B1 \rightarrow\ S1\ \ [\!]\ \ldots\ [\!]\ Bn \rightarrow Sn\ ]$$

Let W be a well-ordering, with a ranking function r: St → W . Let p be an invariant function on the rank r (that is, r(s), where s is the current state) . A helpful direction is one such that an action along that direction decreases the current value of the ranking function (in the well-ordering sense). Define the *helpful directions* in a given state s by a partition of directions into helpful directions HD_s and *steady directions* , STD_s, (instead of the equivalent predicates Q_j -- one for each direction on the states as given in [6]). The helpful directions have the intuitive meaning of being those directions which lead to a positive step in the direction of the desired total correctness property. Under fairness, these directions need not be immediately taken, but will eventually be taken if suitably enabled. This generalizes the notion of always decreasing a well-ordering as defined in the study of total correctness properties without fairness assumptions.

Remark: As we would be considering nonterminating processes(programs), it is possible that the process returns to its initial state- in other words, a state from which it can try to achieve its goal if it wants. For this purpose, we denote the least element of the chain by $\emptyset$ and the highest element by $\top$. For ease of reference, we would refer to the guards as directions.

Fairness of CSP programs: In the case of the nondeterministic selection (as given above), fairness is obviously defined as *fair* selection of the underlying guards. However, in CSP programs which exhibit *local and global nondeterminism,* fairness can be defined at various levels as highlighted in [12]. In this paper, we would be mainly considering essentially process fairness.

Definition 1(Process Fairness): In the case of an infinite execution, each process which is infinitely often enabled, will infinitely often execute. In the CSP case, this can be reformulated as follows: in an infinite execution, if a process is willing to communicate with other processes infinitely often, then the process will do so infinitely often. Note that in the case of CSP the communication is synchronous; that is, the sender and receiver jointly execute the action.

We first use the normal form of CSP as defined in [8]. The language is defined below:

- a program consists of a priori bounded number of processes without shared variables. That is,

$$P::[P_1 \,||\, P_2 \,||\, \cdots \,||\, P_n]$$

- Each process has the following form:

$$P_i \;::\; *[\; \Box_{k=1,n_i}\; B_k^i \;,\; \alpha_k^i \rightarrow S_k^i \;], \text{ where}$$

  - B_k^i is a boolean expression over the local state of P_i;
  - α_k^i is an I/O command of the form $P_j!t$or$P_j?x$ with the usual interpretation occurring in P_i.
  - S_k^i is some loop free and communication free statement in P_i.
  - the loop terminates when all the guards are false considering *distributed termination.*

NOTE: The extension of the work to the complete CSP will be discussed later.

Definition 2: An infinite execution of P is *strongly fair* if and only if every *synchronizable* communication command which is infinitely often enabled is infinitely-often executed. Note that a finite execution is always fair.

Let us now introduce the notion of conspiracy . We can interpret the above definition as follows: If an action is *infinitely often enabled* with all required resources free , it will be infinitely often executed . In other words, the property ensures that all processes that can execute their actions infinitely often will be able to do so eventually. This is referred to as *process allocation fairness*(PAF). However, the notion of PAF is weak and there is a need for more powerful notions for establishing the correctness of concurrent/distributed programs. In the following, we introduce the fairness property of safety from conspiracy. The notion of safety from conspiracy of an action in a concurrent program is defined in terms of the execution sequences. Consider a move a_i of a process P_i and a collection of competing moves $a_{j_1}, \cdots, a_{j_k}$ of processes $P_{j_1}, \cdots, P_{j_k}$.

Definition 3: Processes $P_{j_1}, \ldots, P_{j_k}$ are said to *conspire* against the move a_i of process P_i (i different from $j_1, \ldots, j_k$) if their moves are scheduled so that a_i is never allowed to get enabled (although it would get enabled otherwise).

The above notion of conspiracy has the following significance: in a fair concurrent execution model, we have the possibility that a collection of actions $a_{j_1}, \ldots, a_{j_k}$ (of processes $P_{j_1}, \ldots, P_{j_k}$) may conspire against an action a_i , even under fair allocation of moves (and resources) by the scheduler. For example , this occurs when actions $a_{j_1}, \ldots, a_{j_k}$ alternatively lock up some resources that action a_i needs. In this case, action a_i , though continually or infinitely often enabled in the sequential model , is unable to be executed here due to lack of resources.

Definition 4: The move a_i of process P_i is said to be safe against conspiracy by the moves of competing processes $P_{j_1}, \ldots, P_{j_k}$ iff for all computation sequences t, whenever a_i is infinitely often enabled(that is g_i), along t with respect to each of the competing processes P_i , i.e., if g_i is infinitely often enabled on t , then the move a_i also gets enabled infinitely often along t . Thus, the computation path t does not have an infinite number of occurrences of the moves $a_{j_1},\ldots,a_{j_k}$ without any occurrence of a_i. In other words, if we show that the process cannot be infinitely overtaken, we would be establishing the *starvation freedom* .

The notion of safety from conspiracy is an essential feature of the programs. It becomes important if we are to ensure the proper behaviour of the concurrent programs in the realm of schedulers. Further, the property of fairness with safety from conspiracy for the concurrent execution model is in particular sufficient for ensuring strong fairness in the sequential model (given process fairness of the system). This assumption is also natural in order to resolve the fairness considerations arising out of starvation of moves by conspiracy among processes.

The notion of *conspiracy* is really meaningful in the context really nonterminating programs. In the case of terminating programs (including fair-termination), the notion of conspiracy can be ruled out by invoking the definition of strongly- fair. First we discuss the proof rule as in [8]. Later we discuss the method for establishing the strong fairness of a program in spite of conspiracy. The proof-rule will be referred to as FAIR- TERM.

ASSUMPTIONS:

1. Obviously, we can ignore the guard having the boolean part equal to false. Also, we are interested in showing the termination of the program we can assume that no guard is just equal to true as in such a case the loop does not terminate at all.
2. g_{i_k} would be used to denote the direction in process i; B_{i_k} denotes the boolean part of this direction and α_{i_k} denotes the communication guard of this direction.
3. Let end_i denote the fact that process i has terminated.

2.1 FAIR-TERM: Let W be a well-ordering and let $r_i : S_i \rightarrow W_i$ be the ranking function. The directions for process P_i may then be divided into *helpful directions* HD_i and *steady directions* STD_i in the usual way. Let $\top$ be the maximum element and $\emptyset$ be the minimal element. Let p be the parametrized invariant and GI be the global invariant (cf.[3]).

Note: For the invariance part of the reasoning, one can use any of the partial correctness proof systems([3, 15]).

0. GOAL: GI : $<<p_0>>$ P $<<true>>$

For each $w \in W \neq \phi$ choose a decreasing matching pair of directions g_{i_k} and g_{j_r} such that:

1. **(DEC)**

$$<<p(w) \wedge B_{i_k} \wedge B_{j_r} \wedge w > \emptyset >> B_{i_k};\alpha_{i_k} \mid \mid B_{j_r};\alpha_{j_r} << \exists\ v, v < w \wedge p(v) >>$$

2. **(NOINC)**

$$<<p(w) \wedge B_{i_k} \wedge B_{j_r} \wedge w > \emptyset >> B_{i_k};\alpha_{i_k} \mid \mid B_{j_r};\alpha_{j_r} << \exists\ v, v \leq w \wedge p(v) >>$$

3. **(CONT)**

$$p(w) \wedge w > 0 \rightarrow BB$$

where BB is the disjunction of the boolean part of the matching guards.

4. (TERM)

$$p(\emptyset) \rightarrow \neg BB \ \vee \ (\neg GI))$$

5. (INIT)

$$p_0 \rightarrow \exists \ w \text{ such that } p(w)$$

6. (IOE)

Let $I_m \equiv I \wedge \neg \, (B_{i_k} \wedge B_{j_r})$ for the matching pair. Then

$$I_m: << p(w) \wedge w > 0 >> P << true >>$$

2.2 STRONG FAIRNESS

Consider nonterminating programs. One can observe two features: (a) these could be cyclic programs and (b) after achieving its goal the process does its own computation(i.e. it does not enter into competition anymore). In the context of conspiracy the former is important as the starvation freedom of the latter follows if we establish the deadlock-freedom and ensure that each process will be in the competing section only for a finite time.

Now, we can distinguish basically two types of directions:

1. local (noncritical) and
2. global(critical or competing).

The local directions reflect the fact that process is not interested in any competition whereas the latter are the directions which the reflect the competition of the process with reference to other processes for achieving its goal. The method will be referred to as CONS-FAIR.

CONS-FAIR:

For establishing the starvation-freedom of the program, one has to really establish the following:

Assuming that a process enters its competing states or directions the process enters its goal eventually; in other words, it is not possible for the competing processes to infinitely overtake the process. The establishment of this can be broken down into two steps as given below:

1. Divide the states (also could be considered in terms of the directions of the CSP program as the latter is in its normal form), into local directions and global directions. Now, transform the program by deleting the transitions from the goal (which is the lowest(in the sense of well-founded ordering) direction) to the local states and replacing it by end_i

:= true and adding end_i := false to the initialization part and adding the boolean expression $\neg\, \mathrm{end}_i$ to all the guards. Thus, the program becomes a terminating one once it achieves its goal. For the transformed program, establish that the program terminates using FAIR-TERM.

2. Consider process P_i. Consider any critical direction g_{i_k} . Now, consider the set of processes
$[P_1 \,||\, P_2 \,||\, ...||\, P_{i-1} \,||\, P'_i \,||\, ...\, P_n]$
where P'_i is equal to INIT'; $*[B_{i_k} \rightarrow \text{skip}\ ]$. With reference to this set, establish that there is a deadlock in the transformed program involving the competing processes, say CONS_i, with reference to this direction. In other words, the process gets stuck. This is established for each of the critical directions(or states).

The second part though looks formidable, it is really not as the finding of the ordering in the first part can be used largely in establishing (2).

2.3 EXAMPLE

Obviously, one of the natural examples would be the dining philosophers problem. Most of the solutions are deadlock-free but are not starvation free; that is, processes can conspire so that another process cannot get to eat after it becomes hungry. For lack of space we discuss the algorithm informally. The structure of a deadlock-free solution is given in the appendix. Consider the usual deadlock-free solution having the process"ROOM". The process "ROOM" ensures that there is at most 4 philosophers in the room- thus avoiding deadlock. Each philosopher goes through the cycle of states: think, hungry, pick a fork(left or right), pick the other fork, eat and think. Using our proof method, we can show that the solution is not strongly-fair; The noncritical state(direction) being "think"; the others are critical states(directions). The goal is "eat". The reason is that step 2 of the method fails - inspite of one philosopher being in one of the critical directions the others can proceed to their goals indefinitely. One of the ways of overcoming starvation issues in the dining philosophers problem is to introduce an extra transition for each philosopher after eating-- let us call the new state as *just finished eating*(this being another critical direction). The idea is that the philosopher will enter another state after eating and waits if there are neighbors who are waiting to get hold of the fork for the purpose of eating. A solution on these lines has been given in [13] and [14]. For this solution, we can formally show that the solution is strongly fair. A full analysis will be given in the full paper.

3. DISCUSSION

In this paper, we have given a method for establishing the strong-fairness of CSP programs. The method looks for local and global or competing directions and uses the fair termination of CSP programs under serialized semantics and strong-fairness is established after establishing the fair-termination of the transformed programs and also after showing that if a competing process is stuck in one of the competing directions then at some time in future the other competing processes get stuck(in a sense deadlock) as long as the process remains stuck. The deadlock-detection schemes and the extension to general CSP programs can be done using the techniques given in [1,2]. A complete formal treatment is under way and also techniques of considering failures(bounded) are being investigated and will be reported in the full paper.

REFERENCES

[1] K.R.Apt *Proving Correctness of CSP Programs - a Tutorial,* in Control Flow and Data Flow: Concepts of Distributed Programming(Ed M.Broy),Springer Verlag, Berlin, 1985.

[2] K.R.Apt and N.Francez *Modeling the distributed termination of CSP,* ACM TOPLAS, 6, 3 ,pp 370-379, 1984.

[3] K.R.Apt, N.Francez, W.P. de Roever *A proof system for communicating sequential processes,* ACM TOPLAS 2, 3, pp 359-385, 1980.

[4] K.R.Apt and E.R.Olderog *Proof rules and transformations dealing with fairness,* Science of Computer Programming, 3, 1983.

[5] R.J.R. Back , R. Kurki-Suonio *Serializability in Distributed Systems with Handshaking,* TR CMU-CS-85-109 ,Pittsburgh, 1985.

[6] N. Francez , **Fairness** , manuscript , 1983.

[7] O.Grumberg, N.Francez and S. Katz *A complete proof rule for strong equi-fairness,* LNCS , 164 , 1983.

[8] O. Grumberg, N. Francez and S. Katz *Fair Termination of Communicating Processes* , Proc ACM Symposium on Principles of Distributed Computing, Van Couver, 1984.

[9] O.Grumberg, N.Francez, J.A. Makowsky and W.P. de Roever *A proof rule for fair termination of guarded commands* Proc.of Int. Symposium on Algorithmic Languages , Amsterdam , North-Holland Publishing Co., 1981.

[10] D.Lehmann, A.Pnueli and J. Stavi *Impartiality, Justice and Fairness :the ethics of concurrent termination,* Proc. 8th ICALP , LNCS, 115, Springer-Verlag, 1981.

[11] A. Pnueli *On the extremely fair treatment of probabilistic algorithms* , ACM- Sigact Symposium on Theory of Computing, Boston, 1983.

[12] R. Kuiper and W.P. de Roever *Fairness assumptions for CSP in a temporal logic framework* in D. Bjorner(Ed), *Formal Description of Programming Concepts-II, North-Holland Publishing Co., pp 159-167, 1983.*

[13] S.Mahadevan, and R.K.Shyamasundar On the fairness of distributed programs , Proc. 2nd Conf. on Foundations of Software Technology and Theoretical Computer Science, Bangalore, India, December 1982.

[14] H.Wedde *A starvation free solution of the dining philosophers problem by the use of interaction systems,* Proc. MFCS 81, LNCS 118, pp534-543,1981.

[15] J. Zwiers, W.P. de Roever and P. van Emde Boas *Compositionality and Concurrent Networks :soundness and completeness of a proof system* , ICALP 85, LNCS ,194, Springer Verlag , pp 509-519, 1985

Appendix

A Deadlock-Free Solution for the Dining Philosophers Problem

```
P[i] :: s:=think; pickleft:=false;pickright:=false;

     *[s =think ; room ! enter() → s:=hungry

      ▯ s=hungry; ¬ pickleft; fork(i-1) ? fork() → pickleft:=true

      ▯ s=hungry; ¬ pickright; fork(i) ? fork() → pickright:=true

      ▯ s=hungry; pickleft;pickright → EAT; s:= finished

      ▯s=finished; pickleft; fork(i-1) ! fork() → pickleft:=false

      ▯s=finished; pickright; fork(i) ! fork() → pickright:=false

     ▯ s= finished; ¬ pickleft;¬ pickright;room ! exit() → s:=think
     ]

room::  N:=0;

    *[$0 \leq N < 4$ ; $P_1$ ? enter() → N:=N+1
        ....
     ▯ $0 \leq N < 4$ ; $P_5$ ? enter() → N:=N+1

      ▯ $P_1$ ? exit() → N:=N-1
         ....
      ▯ $P_5$ ? exit() → N:=N-1
     ]

Fork[i] :: picked:=false;

      *[ ¬ picked; $P_i$ ! fork() → picked:=true

       ▯ ¬ picked; $P_{i+1}$ ! fork() → picked:=true

       ▯  picked; $P_i$ ? fork() → picked:=false

       ▯  picked; $P_{i+1}$ ? fork() → picked:=false
       ]
```

ON GLOBAL VS. LOCAL STATE FAIRNESS

RODNEY R. HOWELL, LOUIS E. ROSIER, AND HSU-CHUN YEN*

Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712

ABSTRACT

In this paper, we first discuss the issue of global vs. local versions of state fairness for concurrent programs. We then investigate the complexity of the fair nontermination problem. It turns out that the problem is PSPACE-complete with respect to global state fairness, whereas it is EXPTIME-complete with respect to local state fairness.

1. INTRODUCTION

In recent years, nondeterministic models, such as Petri nets, concurrent programs, communicating sequential processes (CSPs), and networks of communicating finite state machines (CFSMs), have been successfully used to represent many real-world systems. (See, e.g., [1, 9, 15, 18].) A major benefit of such (nondeterministic) models is that they often have a simplier structure than their deterministic counterparts. (cf. [13, 18, 19].) On the other hand, due to the nature of nondeterminism, it is, in general, hard to analyze such systems. One of the fundamental issues concerning nondeterministic systems is the study of problems concerning their infinite behavior (e.g., the deadlock, lockout, starvation, and termination problems), especially when a "fairness constraint" is also taken into account. For the last decade, several notions of fairness for various models have been proposed for problems related to nondeterministic computations. In Carstensen and Valk [2], fairness was used to define languages of Petri nets. They also investigated the well-known dining philosophers problem, taking fairness into consideration. Problems concerning fair termination of concurrent systems can be found in Hart, Sharir, and Pnueli [8], Lehmann, Pnueli, and Stavi [12], Pnueli [16], Rosier and Yen [21], and Vardi [23]. In each of the above, the problem of determining whether a given system of concurrent programs will terminate, under the assumption of some fairness constraint, was investigated. Definitions of fairness concerning networks of CFSMs can be found in Gouda and Chang [6]. A hierarchy of fairness constraints with respect to CSPs was proposed by Manna and Pnueli [14].

In the study of nondeterministic systems, a very useful practice is to represent each system configuration by a "state." A state can be regarded as the result of taking a "snapshot" of the system at a given instant. Such a representation allows us to describe a concurrent system by a directed graph, in which each node represents a system state and each edge indicates a transition (operation) of some program. As far as a single-program nondeterministic system is concerned, this definition of "state" is unambiguous. However, when a system with concurrency is involved (i.e., a

---

[*] H. Yen's current address is: Department of Computer Science, Iowa State University, Ames, IA 50011.

system consists of several programs running in parallel), "states" can mean either "global" or "local" states. A global state is a description of the entire system at some point in its execution. In contrast, a local state (for a particular program) contains only the information concerning the resources to which that program has access. So far, most of the research in concurrent systems adopts the concept of global states. One of the reasons is that global state spaces appear to allow us to have a clean view of the actions within in a system. Furthermore, verification and specification methods for concurrent systems under a global state space representation have been well investigated in recent years. However, as we shall see later, there are some cases for which a global state space fails to capture the real meaning of a concurrent computation. (Sometimes the notion of global state is, in fact, meaningless.) Another drawback is that verification methods concerning global states have implementation difficulties for distributed systems from the practical point of view. Consequently, it is worth taking a closer look at the issue of "global" vs. "local" from both theoretical and practical points of view. The main contribution of this paper is to examine this issue from the theoretical point of view. To achieve this goal, we investigate the global and local versions of *state fairness* with respect to the nontermination problem of concurrent programs. Some related issues of local vs. common (global) knowledge in distributing computing can be found in Kanellakis and Smolka [10], Fagin, Halpern and Vardi [5], Halpern and Moses [7], and Lehmann [11]. See also Reif [20] for related problems concerning games.

Among the various notions of fairness, the state fairness (or an enhanced version called *extreme fairness*) of Pnueli [16] is of great interest because, as was shown in [16], it can be used to capture the essence of "probabilistic" computations. Informally speaking, a computation is said to be *state fair* iff whenever a transition from a state occurs infinitely often, all enabled transitions from that state must be executed infinitely often. The motivation is that if a state occurs infinitely often, and, at each juncture, one tosses an unbiased coin to determine the next transition, then the probability of almost always neglecting some transition (from that particular state) is zero. A similar concept has also been considered by Queille and Sifakis [17] (under the name of *fair choice from states*) to study transition systems and their fairness-related properties. Now, to understand the difference between global and local versions of state fairness, consider the example in Table 1.

```
                         x=y=z=0

  Program A                    Program B

  *[ a1: true → y:=1           *[ b1: true → y:=0; z:=0
  □                            □
  a2: y=2 → stop]              b2: y=1∧z=0 → y:=0; z:=1
                               □
                               b3: y=1∧z=1 → y=2; stop]
```

Table 1: A concurrent system.

Here two programs A and B are running concurrently using y as a shared variable (x and z are A's and B's private variables, respectively). A global state is a tuple (x,y,z); while local states of A and B are represented by (x,y) and (y,z), respectively. Now consider the following infinite sequence σ:

$$[(0,0,0) \xrightarrow{a_1} (0,1,0) \xrightarrow{a_1} (0,1,0) \xrightarrow{b_1} (0,0,0) \xrightarrow{a_1} (0,1,0) \xrightarrow{b_2} (0,0,1) \xrightarrow{b_1} (0,0,0) \xrightarrow{b_1} (0,0,0)]^{\omega}.$$

In this sequence transition a_1 occurs infinitely often in A's two local states (0,0) and (0,1). Also, transition b_1 (b_2) occurs infinitely often in B's local states (0,0), (0,1) and (1,0) ((1,0)). As a result, the sequence is, in a sense, *locally* fair. However, it is not *globally* fair in the sense that a_1 never took place in the global state (0,0,1), although a_1 is enabled infinitely often in that state. It is reasonably easy to see that there is no globally fair infinite computation for such a system. This example reveals that the answer of the fair termination problem of concurrent systems might rely on the underlying notion of fairness (i.e., whether global or local fairness is assumed).

As far as we know, no efforts have been made to clarify the fundamental issue of global vs. local fairness. In fact, the underlying architecture of concurrent systems appearing in the literature is usually based on the concept of a global state space. Although for some applications such an assumption is strong enough, for others it may not be suitable. In addition to this, since each globally fair nonterminating computation is also locally fair, proving that a system will terminate with respect to local fairness will guarantee the termination of the system in terms of global fairness. In what follows, we pose some evidence reflecting the fact that in some cases, local fairness seems to be a better choice:

- In a symmetric system, the computation consists of several identical programs (or processes) running concurrently. It was recently suggested by Clarke, Grumberg and Browne [4] that in some cases symmetric systems can be verified more efficiently than their non-symmetric counterparts. The claim is that in order to verify an N-process symmetric system, it suffices to consider a reduced system with one or two of the processes, and the original system is correct iff the reduced system is correct. Now, to apply the notion of state fairness to such systems and, at the same time, be able to use the reduction technique, it seems to us that local fairness is the more natural choice. (Global fairness does not seem to make a lot of sense here because the meaning of a "global" state of N processes simply disappears after the reduction.)
- Nowadays, one of the trends in designing operating systems, computer networks, and data bases is to make the system distributed. An important feature of distributed systems is that there is no central control which can supervise the entire computation of the system at any time. In addition, the notion of global clock is meaningless for such systems. To implement a distributed system, it is sometimes required to have an on-the-fly verification mechanism (such as time-out). Under such a consideration, global fairness can not be verified (or implemented) in a regular sense because global states of a distributed system can not be recorded unless a "supervising" process is used on top of the system. This is, in some cases, not feasible. For example, in message-passing systems such as networks of CFSMs, a machine can not "freeze" the computation of the system and then acquire the necessary information. Under such a framework, global fairness fails to provide a meaningful notion of fairness. (See Gouda and Chang [6] for several types of local fairness for networks of CFSMs.)
- In many concurrent systems, the ratio of execution speeds between processes can be arbitrarily large (but finite). In this case, for some process the lack of knowledge of the global system may preclude certain events from happening. To see this, consider the

example in Table 1. Suppose the speed ratio of A to B is exactly reflected by the computation σ. In this case, transition a_1 was never ready when the global state (0,0,1) was reached (i.e., when B's private variable z had a value of 1). Therefore, such a "collaboration" will prevent any terminating computations. As a result, state fair computations for such a system might be those defined by local fairness.

Based on the above observations, local vs. global fairness is an issue that deserves further study from both theoretical and practical points of view. In the remainder of this paper, we investigate the complexity of the fair nontermination problem in terms of global and local versions of state fairness. It turns out that the globally fair termination problem is complete for PSPACE, whereas the locally fair termination problem is complete for EXPTIME. This result seems to indicate that despite the above merits for some applications, problems related to local fairness are, in general, harder to analyze than those related to global fairness. We also hope that the results of this study will allow us to have a better insight into the nature of nondeterministic computations.

2. THE MODEL

A (finite-state) ***nondeterministic program*** (or *program* for brevity) P is a tuple (Q,V,δ,X,q_0), where

- Q is the finite set of program locations,
- $V=\{v_1,\ldots,v_h\}$ is a finite set of Boolean variables,
- $\delta: Q\times\{0,1\}^h \rightarrow 2^{Q\times\{0,1\}^h}$ is the transition function,
- $X\subseteq Q$ is the set of terminal program locations,
- $q_0 \in Q$ is the initial program location.

In this paper, high-level language constructs, such as **if**...**then**...**else**, will be used to define δ (although we will not define the formal syntax of these language constructs). This kind of abstraction allows us to describe δ in a more easily understood manner, thus making programs more readable. We use s[...]s′ to denote that upon executing the transition [...] at location s, the program will enter location s′ (the contents of the variables will also be changed accordingly). The *size* of P, denoted by $|P|$, is defined to be the length of the description of P when a standard binary encoding technique is used. A system of k concurrent programs S is a tuple $(P_1,\ldots,P_k)$, where each $P_i=(Q_i,V_i,\delta_i,X_i,q_{i,0})$, $1 \leq i \leq k$, is a program. The size of S, denoted by $|S|$, is defined to be $|P_1|+\ldots+|P_k|$. For every $1 \leq i,j \leq k$ $(i \neq j)$, $G_{i,j}=V_i \cap V_j$ is the set of *shared* variables between P_i and P_j. $V_i-G_{i,j}$ is the set of *private* variables of P_i *with respect to* P_j. A set of variables V is *local* to P_i iff $\forall\ V_j$, $i \neq j$, $V \cap V_j=\emptyset$. For a variable v, let $\bar{v}$ denote the value of v. A *local* state for program P_i is a tuple $(q, <\bar{v}_1,\ldots,\bar{v}_{h_i}>)$ (or $(q, \overline{V_i})$), where $q \in Q_i$ and $V_i=\{v_1,\ldots,v_{h_i}\}$. A *global* state for system S is a tuple $[(q_1,\overline{V_1}),\ldots,(q_k,\overline{V_k})]$, where $(q_j,\overline{V_j})$ is a local state of P_j, for $1 \leq j \leq k$. A transition t (of P_i): $(q,s) \rightarrow (q',s')$ is *enabled* in a local state (q,s) iff $(q',s') \in \delta_i(q,s)$. t is *enabled* in a global state q $(=[q_1,\ldots,q_k]$, where each q_i is a local state of program $P_i)$ iff t is enabled in some local state q_i. A *computation* is a sequence of global states

$$\sigma\colon [q_1^0,\ldots,q_k^0]\text{-}t_1 \rightarrow [q_1^1,\ldots,q_k^1] \cdots \text{-}t_i \rightarrow [q_1^i,\ldots,q_k^i]\text{-}t_{i+1} \rightarrow [q_1^{i+1},\ldots,q_k^{i+1}] \cdots \rightarrow \cdots,$$

which satisfies:

- q_j^0, $1 \leq j \leq k$, is the initial state of P_j,
- $\forall\ i \geq 0$, $\exists m$, $1 \leq m \leq k$, such that $t_i\colon q_m^i \rightarrow q_m^{i+1}$ ($\in\ \delta_m$) and for all j, $j \neq m$, the contents of private variables of P_j with respect to P_m are unchanged.

A computation can be finite or infinite. A transition t is *globally fair* (*locally fair*) for a computation σ iff t occurs infinitely often from all the global (local) states at which it is enabled infinitely often. A computation is globally fair (locally fair) iff all of its transitions are globally fair (locally fair). Note that a globally fair computation is also a locally fair computation. A *nonterminating* computation is a computation in which no program enters a terminal program location. The *globally* (*locally*) *fair nontermination problem* is to determine, given a system of concurrent programs, whether there exists a globally (locally) fair nonterminating computation.

3. THE COMPLEXITY RESULTS

In this section, we show that the globally (locally) fair nontermination problem is complete for PSPACE (EXPTIME). This suggests that despite the aforementioned merits, problems related to local fairness are, in some cases, harder to analyze than those related to global fairness.

Theorem 1: The globally fair nontermination problem is PSPACE-complete.

Proof sketch. First we show that the problem is in PSPACE. Let $S=(P_1,\ldots,P_k)$ be a k-program system of size n. Clearly, the number of distinct global states is bounded by 2^n. Now, for easy understanding we consider a directed labelled graph G_S such that each node represents a global state, and an edge labelled t from node s to s′ indicates that transition t can take the system from state s to s′. (This graph will not be constructed because it requires more than a polynomial amount of space.) Let s_0 be the node corresponding to the initial state of the system. Given two nodes s and s′, let $s \rightarrow s'$ denote that there is a path from s to s′. It is not hard to see that there will be a globally fair nonterminating computation iff there exists a subgraph G'_S such that

(1) G'_S is a strongly connected component,

(2) $\exists\ s \in G'_S$ such that $s_0 \rightarrow s$,

(3) no node in G'_S is a terminal state,

(4) $\forall$ transition t, if t is globally enabled in a state s in G'_S, then there exists an edge (s,s′) labelled t, for some s′ in G'_S.

Now, note that given two nodes s and s′, "Is $s \rightarrow s'$?" and its complement "Is $s \not\rightarrow s'$?" can be detected in DSPACE(n^2). (In each case, a 2-alternation, O(n)-space bounded alternating Turing machine (ATM), as defined by Chandra, Kozen, and Stockmeyer [3], can be built to test the condition. As proved in [3], the complexity result for such an automaton follows from a generalization of Savitch's theorem [22].) The decision procedure for the globally fair termination problem invloves, first, guessing a node s in G'_S, and checking that $s_0 \rightarrow s$. (It takes O(n) bits to represent the node s.) Then, to make sure that the rest of the aforementioned conditions hold, one can do the following test:

```
begin
    for each $s_i \in G_S$ do
        begin
            if $s \rightarrow s_i$ and $s_i \not\rightarrow s$ then reject
            if $s \rightarrow s_i$ and $s_i$ is a terminal state then reject
        end
end
```

To implement this procedure, one needs to use an n-bit counter to keep track of the value i. Consequently, the globally fair nontermination problem is in PSPACE.

The PSPACE-hardness result is obtained by reducing the LBA acceptance problem, which is well-known to be PSPACE-complete, to the globally fair nontermination problem. Given an arbitrary deterministic linear bounded automaton (DLBA) M and an input x, we show how to construct a program P to simulate the computation of M on x in such a way that M accepts x iff P will not terminate with respect to the global state fairness constraint. The lower bound then follows. Without loss of generality, we can assume that M always halts. The contents of the worktape of M are simulated by an array of Boolean variables of P. At any instant, the current program location of P represents the current state and the current head position of M. The simulation proceeds step by step until an accepting state of M is reached. Then, P enters an infinite loop which corresponds to a portion of the globally fair nonterminating computation. M accepts x iff P has a globally fair nonterminating computation. The details are left to the reader. □

Observe that for single program systems the notions of local and global state fairness are identical. In what follows, we show that the locally fair nontermination problem is EXPTIME-complete for a system of more than one program. The lower bound result is obtained by showing that a concurrent system can be used, in some sense, to simulate the computation of an arbitrary linear space bounded ATM. Since it was shown in [3] that such automata have the same computational power as that of exponential time bounded deterministic Turing machines, the lower bound result follows.

Theorem 2: The locally fair nontermination problem for a k-program system, $k \geq 2$, is EXPTIME-complete.

Proof sketch. First, we show the lower bound result. In what follows, given a linear space bounded ATM M and an input x, we show how to construct a concurrent system $S=(P_1,P_2)$ of two programs to simulate the computation of M on x in such a way that M accepts x iff S has a locally fair nonterminating computation. Without loss of generality, we assume that the number of transitions from any state in M is at most 2. (If not, an equivalent ATM, which satisfies this requirement, can be built easily.)

Basically, P_1 repeatedly simulates the entire computation of M on x while P_2 controls nondeterministic choices. The simulation proceeds in phases corresponding with the phases (universal or existential) of M. When the simulation is in a universal phase, program P_2 is disabled. During this phase, since both outgoing transitions (if they exist) from a state, say q, of P_1 are enabled simultaneously, for a state fair computation both transitions (from q) must be executed infinitely often if q occurs infinitely often. In other words, the fairness constraint ensures a valid simulation of universal moves. Now, when the simulation reaches an existential phase, P_2 will set a flag (a shared variable), say f, to 0 or 1. P_1 will then choose one of its two transitions to execute (if they exist) based on the value of f. The reason P_2 sets f is to allow one and only one of P_1's transitions (if they exist) to be enabled. Such a mechanism ensures a correct simulation of existential moves. Finally, once the simulation of P_1 reaches an accepting state of M, the entire procedure will be restarted, and P_2 again sets f to either 0 or 1 (P_1 does not use this value of f; rather, this step is included in order to ensure that the computation is locally fair for P_2). It is then reasonably easy to see that M accepts x iff there is a locally fair nonterminating computation. It is worth mentioning that such a fair computation is not necessarily globally fair. The reason is that in order to simulate some existential move, P_2 must be allowed to set f to the same value each time through the computation. The rest of the details are left to the reader.

Now, we will show the upper bound. In a system of concurrent programs of size n, the number of global states is bounded by $O(2^n)$. Hence, one can construct a directed graph of size $O(2^n)$ such that each node represents a global state and each edge indicates a transition of some program. To determine whether a locally fair nonterminating computation exists, one needs to consider only those strongly connected components that can be reached from the initial node in the graph representation of the system, and for each such component, one then checks the conditions of being locally fair. More precisely, if $[q_1,\ldots,q_i,\ldots,q_k]$ is a node (a global state) in the component and t is a transition enabled in q_i, $1 \leq i \leq k$, then there must exist two nodes $[r_1,\ldots,q_i,\ldots,r_k]$ and $[r'_1,\ldots,q'_i,\ldots,r'_k]$ such that the subgraph $[r_1,\ldots,q_i,\ldots,r_k] \xrightarrow{t} [r'_1,\ldots,q'_i,\ldots,r'_k]$ is in the component. For a graph of size $O(2^n)$, the above procedure can be done in $O(2^n)$ steps. The details are left to the reader. (See [21] for a similar proof.) □

References

[1] Bochmann, G. and Sunshine, C., Formal methods in communication protocol design, *IEEE Trans. Commun.* (1980), pp. 624-631.

[2] Carstensen, H. and Valk, R., Infinite behaviour and fairness in Petri nets, *Proc. of Advances in Petri Nets LNCS 188* (1985), 83-100.

[3] Chandra, A., Kozen, D. and Stockmeyer, L., Alternation, *JACM 28*, 1 (January 1981), pp. 114-133.

[4] Clarke, E., Grumberg, O. and Browne, M., Reasoning about networks with many identical finite-state processes, to appear in *Proc. of 5th ACM Symp. on Principles of Distributed Computing,* 1986.

[5] Fagin, R., Halpern, J. and Vardi, M., A model-theoretical analysis of knowledge, *Proc. of 25th IEEE Symp. on Foundations of Computer Science* (1984), pp. 268-278.

[6] Gouda, M. and Chang, C., Proving liveness for networks of communicating finite state machines, *ACM Trans. on Programming Languages and Systems 8,* 1 (1986), pp. 154-182.

[7] Halpern, J. and Moses, Y., Knowledge and common knowledge in a distributed environment, *Proc. of 3rd ACM Symp. on Principles of Distributed Computing* (1984), pp. 50-61.

[8] Hart, S., Sharir, M. and Pnueli, A., Termination of probabilistic concurrent programs, *ACM Trans. on Programming Languages and Systems 5,* 3 (July 1983), pp. 356-380.

[9] Hoare, C.A.R., Communicating sequential processes, *CACM 21* (1978), pp. 666-677.

[10] Kanellakis, P. and Smolka, S., On the analysis of cooperation and antagonism in networks of communicating processes, *Proc. of 4th ACM Symp. on Principles of Distributed Computing* (1985), pp. 23-38.

[11] Lehmann, D., Knowledge, common knowledge and related puzzles, *Proc. of 3rd ACM Symp. on Principles of Distributed Computing* (1984), pp. 62-67.

[12] Lehmann, D., Pnueli, A and Stavi, J., Impartiality, justice and fairness: The ethics of concurrent termination, *Proc. of 8th International Colloquium on Automata, Languages and Programming,* LNCS 115 (1981), pp. 264-277.

[13] Lehmann, D. and Rabin, M., On the advantages of free choice: a symmetric and fully distributed solution to the dining philosophers problem, *Proc. of the 10th ACM Symp. on Principles of Programming Languages* (1981), pp. 133-138.

[14] Manna, Z. and Pnueli, A., How to cook a temporal proof system for your pet language, *Proc. of 10th Annual ACM Symp. on Principle of Programming Languages* (1983), pp. 141-154.

[15] Peterson, J., *Petri net theory and the modeling of systems,* (Prentice-Hall, Englewood Cliffs, NJ, 1981).

[16] Pnueli, A., On the extremely fair treatment of probabilistic algorithms, *Proc. of the 15th Annual ACM Symp. on Theory of Computing* (1983), pp. 278-290.

[17] Queille, J. and Sifakis, J., Fairness and related properties in transition systems - a temporal logic to deal with fairness, *Acta Informatica 19* (1983), pp. 195-220.

[18] Rabin, M., N-process synchronization by $4^*\log_2 N$-valued shared variable, *Proc. of the 21st Annual Symp. on Foundations of Computer Science* (1980), pp. 407-410.

[19] Rabin, M., The choice coordination problem, *Acta Informatica 17* (1982), pp. 121-134.

[20] Reif, J., Universal games of incomplete information, *Proc. of 11th ACM Symp. on Theory of Computing* (1979), pp. 288-308.

[21] Rosier, L. and Yen, H., On the complexity of deciding fair termination of probabilistic concurrent finite-state programs, *Proc. of 13th International Colloquium on Automata,*

Languages and Programming, LNCS 226 (1986), pp. 334-343.

[22] Savitch, W., Relationships between nondeterministic and deterministic tape complexities, *J. of Computer and System Sciences 4*, 2 (1970), pp. 177-192.

[23] Vardi, M., Automatic verification of probabilistic concurrent finite-state programs, *Proc. of the 26th Annual Symposium on Fundations of Computer Science* (1985), pp. 327-338.

THE MEMBERSHIP AND EQUIVALENCE PROBLEMS FOR PICTURE LANGUAGES[1]

CHANGWOOK KIM
School of Electrical Engineering
and Computer Science
The University of Oklahoma
Norman, Oklahoma 73019

IVAN HAL SUDBOROUGH
Computer Science Program
University of Texas at Dallas
Richardson, Texas 75083

ABSTRACT

The paper describes two principal results: (1) there is a nondeterministic polynomial time algorithm to determine whether an arbitrary picture is a member of a context-free picture language and (2) the equivalence problem for regular picture languages is not decidable (not even partially decidable). This solves open questions described in earlier publications. Furthermore, these results are best possible. That is, the membership problem for context-free picture languages was earlier shown to be NP-hard and the complement of the equivalence problem for regular picture languages is easily seen to be partially decidable.

1. INTRODUCTION

In [SW] it was shown that the membership problem for regular picture languages is NP-complete and that the emptiness of intersection problem for regular picture languages is undecidable. It was also shown that the membership problem for context-free picture languages can be solved in polynomial space and that there exists a set of pictures P described by a context-free language L such that, for every integer n, there is a picture p in P of size n for which the shortest string in L that draws p has length at least 2^n. (The latter implies that the membership problem for context-free picture languages cannot be solved in polynomial time by the simple expedient of (1) guessing a string x in the language and (2) verifying that x draws the given picture. Since the string x may have exponential length, this process is not polynomial time bounded.) Whether the equivalence problem for regular picture languages could be solved by an algorithm and whether the membership problem for context-free picture languages is in the class NP were left as open questions.

---

[1] A portion of this work is contained in a Ph.D. dissertation submitted to the Electrical Engineering and Computer Science Department at Northwestern University by the first author.

We describe solutions to these problems. We show that the membership problem for context-free picture languages is, indeed, in the class NP and consequently is NP-complete. We also show that the equivalence and containment problems for regular picture languages are not decidable.

We shall assume the reader is familiar with the basics of formal language and automata theory [HU]. For the 'picture-part', we informally discuss necessary notations. For more detailed definitions regarding picture descriptions by strings, we refer to [MRW] or [SW].

Consider a four letter alphabet $\pi = \{u,d,r,l\}$. Each string x over the alphabet π describes a picture in the sense that it describes a walk in the two-demensional Cartesian plane, where u,d,r, and l represent movements up, down, right, left, respectively. For a string w in π, the drawn picture described by w, denoted by dpic(w), is a triple $\langle p,s,e\rangle$, where p is a picture, s is the start-point, and e is the end point. If dpic(w) = $\langle p,s,e\rangle$, then the picture described by w, denoted by pic(w), is p. Figure 1.1 illustrates examples of a drawn picture and a picture described by the string $w = ur^2uld$. (Note: a small circle or square in a picture denotes a start or end point, respectively.)

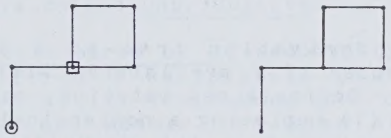


Figure 1.1. Examples of a drawn picture and a picture.

For a given language L over the alphabet π, the set of drawn pictures described by L, denoted by dpic(L), is dpic(L) = {dpic(w) | w is in L}. Similarly, the set of pictures described by L, denoted by pic(L), is pic(L) = {pic(w) | w is in L}.

A set of pictures (or drawn pictures) P is regular if there is a regular set R such that dpic(R) = P. Similarly, a set of pictures (or drawn pictures) P is context-free if there is a context-free grammar G such that dpic(L(G)) = P.

2. MEMBERSHIP COMPLEXITY FOR CONTEXT-FREE PICTURE LANGUAGES

We show in this section that the picture membership problem for context-free languages is in the class NP. (The picture membership problem for a context-free language L asks, for a given picture p, whether p is in pic(L) or not.) From this it follows by Lemma 2.1 that the picture membership problem for context-free languages is, in fact, NP-complete.

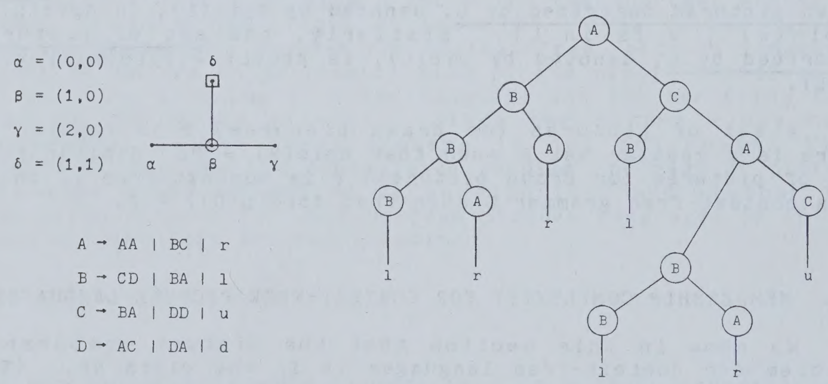
Lemma 2.1 (Sudborough and Welzl [SW]) The picture membership problem for regular picture languages is NP-complete.

Lemma 2.2 (Sudborough and Welzl [SW]) Let $G=(N,\pi,P,S)$ be a context-free π-grammar in Chomsky Normal Form. For every picture p in pic(L(G)), there is a string w in L(G) and a derivation tree of w in G with depth at most $4\cdot|N|\cdot|p|^3$ such that p=pic(w).

Sudborough and Welzl also showed that there is a context-free language L and, for each positive integer n, a picture p_n such that (1) p_n is in dpic(L) and (2) if w is in L and p=dpic(w), then w has length at least $2^{|p|}$. Consequently, the straightforward approach: (1) guess a string w, (2) verify that w is in L(G), and (3) verify that p=dpic(w), takes exponential time.

We describe a polynomial time process which does not explicitly generate a string w in L(G). We generate a derivation tree for a string w with some of the tree pruned away and show that, for any picture p in pic(L), there is always such a pruned tree with size polynomial in the size of the picture p. Thus, we construct a nondeterministic algorithm which, when given a picture p, guesses a tree and verifies that it is a pruned derivation tree for a string w such that p = pic(w).

Let T be a derivation tree of a string w and let p=dpic(w). The nodes of T are labeled with nonterminal and terminal symbols. Define a new labeling, called the augmented labeling of T, by (1) replacing a nonterminal label A on a node x with a label (A,s,e), where (a) the subtree of T with x as a root describes a derivation $A \Longrightarrow^* v$ and (b) in the drawing of p described by the string w the substring v draws a subpicture of p starting at position s and ending at position e and (2) leaving the terminal symbol labels on nodes of T as they were. See Figures 2.1-2.2.



α = (0,0)
β = (1,0)
γ = (2,0)
δ = (1,1)

A → AA | BC | r
B → CD | BA | l
C → BA | DD | u
D → AC | DA | d

Figure 2.1. A picture, a context-free grammar, and a derivation tree.

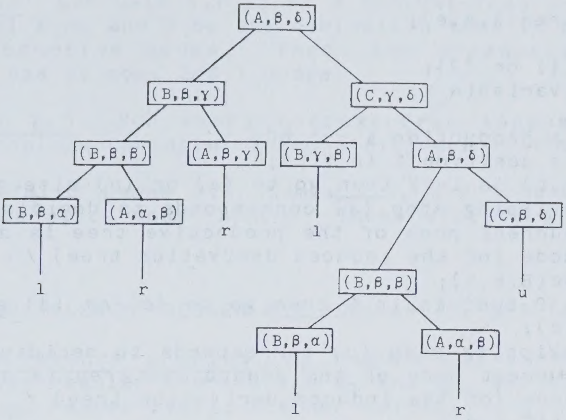


Figure 2.2. Augmented labeling of a derivation tree.

Consider the linear ordering of the nodes of T created by a postorder traversal of T. A node x of T is productive if it is not a leaf of T and there is no node y in T such that (1) y<x and (2) x and y have the same augmented label. A node that is not productive is called unproductive.

The productive subtree of T, denoted by prod(T), is the smallest subtree T' of T satisfying: (1) T' contains all productive nodes of T, (2) T'contains the root of T, and (3) each node x of T' has either the same number of direct descendants as node x of T or has no direct descendants, i.e. is a leaf of T'. (A leaf node x of T' which is not a leaf of T is called an external node.)

Given a drawn picture p we wish to know whether there exists a derivation tree T for a string x from the given grammar G such that p=dpic(x). In fact, it is sufficient to produce the productive subtree prod(T) of such a derivation tree T. The algorithm will attempt to produce prod(T) using a standard nondeterministic top-down parsing strategy. Each time the algorithm verifies that a node x with the augmented label (A,s,e) is the root of a subtree generating a string v which draws a portion of the picture p starting from position s and ending at position e, it records the triple (A,s,e) in a set E. When an external node y of prod(T) is encountered, say with augmented label (A,s,e), the algorithm does not generate a subtree with y as a root. It simply checks that the augmented label (A,s,e) is already in the set E. If so, the algorithm knows that an appropriate subtree can be generated and does not explicitly do so. Our algorithm does not know, however, that a particular node is an external node. It simply guesses that a candidate node x with augmented label (A,s,e) is an external node, but it is only allowed to do so when the triple (A,s,e) is in E. We describe now our algorithm:

```
procedure tree(A,s,e);
begin
do one of (1) or (2);
global set variable E;

(1) choose a production A --> BC;
    choose a position t in V(p);
    if (B,s,t) is in E then go to (a) or (b) else go to (a);
        / skipping step (a) corresponds to deciding that the
          current node of the productive tree is an external
          node (of the induced derivation tree) /
    (a) tree(B,s,t);
    (b) if (C,t,e) is in E then go to (c) or (d) else go to
          (c);
        / skipping step (c) corresponds to deciding that the
          current node of the productive tree is an external
          node (of the induced derivation tree) /
    (c) tree(C,t,e);
    (d) E <-- E U {(A,s,e)};

(2) choose a production A --> a (where a is in {r,l,d,u});
    if the position e results from the movement described by
    the symbol a from the position s and the edge (s,e) is in
    the picture p then E <-- E U {(A,s,e)} else terminate
    unsuccessfully;

end.
```

We determine if a candidate drawn picture $\langle p,s,e\rangle$ is in dpic(L(G)) for a given context-free grammar G in Chomsky Normal Form by calling the procedure with tree(S,s,e) and the global set variable E initialized to the empty set, where S is the initial nonterminal of G. The algorithm either (a) verifies the existence of a productive subtree T' of a derivation tree T for a string w such that $\langle p',s,e\rangle$ = dpic(w) and p' is a subpicture of p or (b) it stops unsuccessfully. When the algorithm stops successfully the set E will contain the set of triples used in the augmented labeling of the productive subtree T' it has produced. We can then check to see if the entire picture p is generated by checking for each edge (q,r) in p if there exists a nonterminal symbol B such that (B,q,r) is in E. If so, then the algorithm stops and answers "yes, the picture is in dpic(L)". Otherwise, the algorithm stops with an unsuccessful outcome.

The amount of time to execute the call to the procedure with tree(S,s,e) is proportional to the size of the productive subtree T' that the algorithm generates. We show now that there is always a productive subtree T' whose size is bounded by a polynomial of the number of edges (or vertices) in the picture p. Thus, we show that our nondeterministic algorithm has a polynomial time bound.

Lemma 2.3 Let $G=(N,\pi,P,S)$ be a context-free π-grammar in Chomsky Normal Form, $p = \langle r,s,e\rangle$ be a drawn picture in dpic(L(G)), and x be in L(G) such that dpic(x)=p. If T is a derivation tree of x, then T has at most $|N|\cdot(|p|+1)^2$ productive nodes.

Lemma 2.4 Let $G=(N,\pi,P,S)$ be a context-free π-grammar in Chomsky Normal Form and T be any derivation tree in G of height d with m productive nodes. Then, the productive subtree prod(T) of T has at most 2md+1 nodes.

Theorem 2.5 For every context-free language L, the membership problems for dpic(L) and pic(L) are in the class NP.

The proofs of Lemma 2.3, Lemma 2.4, and Theorem 2.5 are omitted due to space constraints. These will be given in the full paper.

3. THE EQUIVALENCE PROBLEM FOR REGULAR PICTURE LANGUAGES

The equivalence problem for regular picture languages asks, given two regular expressions E_1 and E_2, denoting regular sets L_1 and L_2, respectively, whether $\text{dpic}(L_1) = \text{dpic}(L_2)$. The containment problem with the same input asks whether $\text{dpic}(L_1)$ is a subset of $\text{dpic}(L_2)$. In this section we show that the equivalence and containment problems for regular picture languages are undecidable. This is done by a reduction from the problem of deciding whether or not a given canonical form (oblivious) one-tape Turing machine with three tape symbols, a, b, and the blank symbol B, accepts the empty string. The Turing machine problem is known to be undecidable [HU]. (We assume a tape that is infinite in both directions whose every cell contains the blank symbol B initially, except those containing the input.)

A canonical form (oblivious) one-tape Turing machine is a deterministic one-tape Turing machine that has a predetermined head motion on any input of a given length. A canonical form Turing machine is defined so that on an input string of length n it moves its head right n times, then moves it left n+1 times, then right n+2 times, and so on, each pass increasing by one the number of movements of the head before direction is changed. In fact, let's assume, without loss of generality, that a canonical form Turing machine never prints a blank. Then, a canonical form Turing machine is simply a one-tape Turing machine that always moves its head either left or right, changes its direction of head motion when and only when it scans a blank symbol, and that initially is in the move-head-to-the-right mode.

Let M be such a canonical form one-tape Turing machine. Our reduction constructs regular expressions E_M and E_I such that $\text{dpic}(E_M)$ is a subset of $\text{dpic}(E_I)$ if and only if M does not accept the empty string, i.e., M does not enter a final state when started on a completely blank tape. Let L_M be the set denoted by the regular expression E_M. The pictures in L_M correspond, as we shall see, to sequences of Turing machine movements which have the correct head motion for a computation by a canonical form Turing machine on a blank tape. Furthermore, such pictures are drawn only for sequences of transitions which (1) are consistent with a computation in that the state entered in the i-th step is the same as the beginning state in the (i+1)-st step of the sequence, and (2) terminate with a final state.

The pictures describe such a sequence of transitions by representing successive passes of the head over the written portion of the tape through successive rows in the picture. For all i, the i-th pass will be represented by row 5(i-1) of the picture, where row 0 is taken to be the initial row. The picture describes a sequence of movements in which (1) reading symbol a, b, or B, respectively, is represented by drawing a rectangle of width 1 and height 1, 2 or 3, respectively, above the row, say r, of the current pass such that the top of the drawn rectangle is on the fourth row above r and (2) writing symbol a or b, respectively, is normally represented by drawing a rectangle of width 1 and height 1 or 2, respectively, below the row, say r, of the current pass such that the top of the rectangle is on the first row below row r. However, when the symbol read is the blank symbol B, then the Turing machine reverses the direction of motion of its head and begins its next pass. So, the symbol printed by the Turing machine is not represented by an appropriate size rectangle below the current row r, but is represented by an appropriate size rectangle below row r-5. That is, the rectangle is drawn below the row corresponding to the next pass.

In addition, (1) if the current pass was moving the head from left to right, then we draw, just to the right of the rectangle representing the symbol replacing the blank, a rectangle representing another blank B and (2) if the current pass was moving the head from right to left, then we draw, just to the left of the rectangle representing the symbol replacing the blank, a rectangle representing another blank B. In both cases this corresponds to drawing a blank in the indicated position and is appropriate as the tape contains a blank in all cells initially.

The types of rectangles drawn are indicated in Figure 3.1.

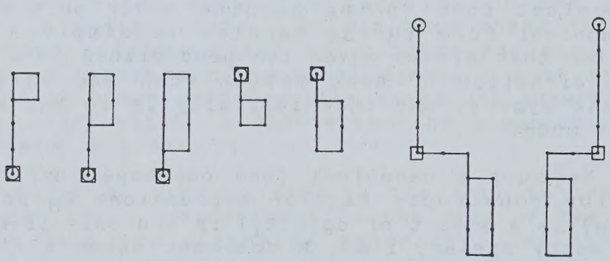


Figure 3.1. Picture components for reading a, reading b, reading B, writing a, writing b, down-left, and down-right (from left to right).

We choose to draw these rectangles in the following way: (1) to draw the unit size rectangle corresponding to reading the symbol a, we use the string r(a) = u^4rdld^3, (2) to draw the size two rectangle corresponding to reading the symbol b, we use the string r(b) = $u^4rd^2ld^2$, (3) to draw the size three rectangle corresponding to reading the blank symbol B, we use the string r(B) = u^4rd^3ld, (4) to draw the unit size rectangle corresponding to writing the symbol a, we use the string w(a) = d^2rulu, and (5) to draw the size two rectangle corresponding to writing the symbol b, we use the string w(b) = d^3ru^2lu. To go down to the next row, draw a size three rectangle corresponding to a blank symbol, and then start a left moving pass, we use a string down-left = $d^5r^2d^4ru^3lul^2$. To go down to the next row, draw a size three rectangle corresponding to a blank symbol, and then start a right moving pass, we use the string down-right = $d^5l^2d^4ru^3lur^2$.

We construct first a right linear grammar $G_M = (N,\pi,P,S)$. We can convert G_M into the desired regular expression E_M such that E_M denotes the language G_M generates. Any of the standard algorithm in the literature can be used [HU]. The nonterminals of G_M are the states of the Turing machine M. Rules for constructing its production are given below (where both c and d represent variables, which represent either the symbol a or the symbol b, but not the blank symbol B).

S ---> $l^2d^4ru^3lur^2$ down-left w(c) l^2 p

if M moves from its initial state p_0, scanning the blank symbol B, to state p, prints c, and moves its head left,

q ---> r(B) down-right w(c) r^2 p

if M moves from state q, scanning the blank symbol B to state p, prints c, and moves its head right,

q ---> r(B) down-left w(c) l^2 p

if M moves from state q, scanning the blank symbol B to state p, prints c, and moves its head left,

q ---> r(d) w(c) r^2 p

if M moves from state q, scanning d, to state p, prints c and moves its head right,

q ---> r(d) w(c) l^2 p

if M moves from state q, scanning d, to state p, prints c and moves its head left, and

q ---> e if q is a final or accepting state.

An example of a picture drawn by a string in the language L_M is shown in Figure 3.2.

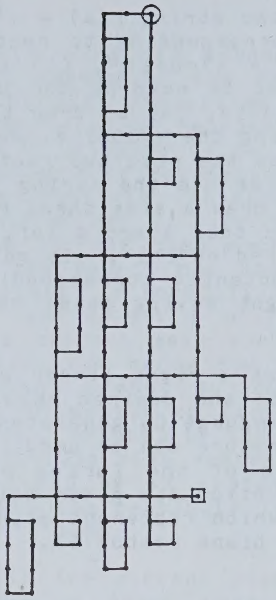


Figure 3.2. An example of a picture in dpic (L_M).

There are many pictures described by strings in L_M, but at most one corresponds to a computation by M on a blank tape. That picture has the property that in each row r (pass of the Turing machine M) the same rectangle is drawn above row r as was drawn below row r+5. That is, the picture corresponds to a computation if and only if the symbol read at each step is the same as the symbol last written on that cell. In other words, a picture does not correspond to a computation if and only if the picture contains (1) a size two rectangle and a size one rectangle in the same position, (2) a size two rectangle and a size three rectangle in the same position, or (3) a size one rectangle and a size three rectangle in the same position. These invalid subpictures are shown in Figure 3.3.

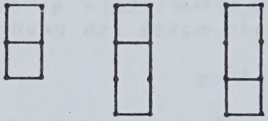


Figure 3.3. Invalid subpictures.

We define a regular expression E_I, which denotes a language L_I such that L_I contains all strings with at least one occurrence of an invalid subpicture described in Figure 3.3. Let I_1, I_2, and I_3 be strings which draw the three invalid subpictures shown in Figure 3.3. Then an appropriate regular expression E_I is the regular expression $\pi^*(I_1 + I_2 + I_3)\pi^*$.

The reader should observe that any picture drawn by a string in L_M that corresponds to a computation (does not contain any of the invalid subpictures) must describe a computation by M on a completely blank tape. This is so, because the initial production of the grammar ensures that a size three rectangle (denoting a blank symbol B) is drawn at the left end of the first row and also ensures that the first transition is on a blank symbol and then changes the direction of motion of its head to move to the left. Furthermore, the Turing machine is constrained to be in canonical form so it changes the direction of motion of its head when and only when the head is scanning a blank. The grammar ensures that everytime the direction of head motion is reversed a size three rectangle (denoting a blank symbol) is drawn at the end of the next row in a position one cell further along than the end of the preceding row. Thus, when the picture corresponds to a computation it must be the computation of M on a completely blank tape.

Theorem 3.1 The containment problem for regular picture language is undecidable. (Note: dpic(L_M) is a subset of dpic(L_I) if and only if M does not accept the empty string.)

The proof of Theorem 3.1 is omitted due to space constraints. This will be given in the full paper.

Corollary 3.2 The equivalence problem for regular picture languages is undecidable. (Note: Let $L = L_M \cup L_I$, then dpic(L) = dpic(L_I) if and only if M does not accept the empty string.)

REFERENCES

[HU] J. E. Hopcroft and J.D. Ullman, Intro. to Automata Theory, Languages, and Computation, Addison-Wesley, Reading, Mass. 1979.

[MRW] H.A. Maurer, G. Rozenberg, E. Welzl, "Using string languages to describe picture languages", Information and Control 54 (1982), pp. 155-185.

[SW] I.H. Sudborough and E. Welzl, "Complexity and decidability for chain code picture languages", Theoretical Computer Science 36 (1985), pp. 173-202.

AVOIDING LATCH FORMATION IN REGULAR EXPRESSION RECOGNIZERS

Michael J. Foster
Computer Science Department
Columbia University
New York City 10027

Abstract

Specialized silicon compilers, or module generators, are promising tools for automating the design of custom VLSI chips. In particular, generators for regular language recognizers seem to have many applications. This paper identifies a problem called latch formation that causes regular expression recognizers to be more complex than they would first appear. If recognizers are constructed in the most straightforward way from certain regular expressions, they may contain extraneous latches that cause incorrect operation. After identifying the problem, the paper presents a ''source-to-source'' transformation that converts regular expressions that cause latch formation into expressions that do not. This transformation allows regular expression recognizers to be simpler and smaller, thus adding to the advantages of specialized silicon compilers.

1. Introduction

One of the most important tools available to today's designers for speeding up the process of designing custom VLSI is the *specialized silicon compiler*. Sometimes called *module generators*, specialized silicon compilers are tools that combine knowledge of a specific application area with a set of primitive cells and rules for combining the cells to produce a layout. Specialized silicon compilers will only produce layouts for their specialized areas of application, but will do a very good job within those areas. The chips that they produce will be small and fast. Specialized silicon compilers have been built for digital signal processors [6, 14], combinational logic [16], data paths [3, 11, 13], general-purpose processors [17], synchronizers [2, 5], and the topic of this paper, pattern matchers [9, 12]. Anything that improves the efficiency of the chips produced by specialized silicon compilers is a worthwhile contribution to the field of VLSI design automation.

This paper discusses specialized silicon compilers that produce recognizers for regular expressions. A regular expression recognizer is a pattern matching circuit in which the pattern is specified by a regular expression. A specialized silicon compiler for these recognizers will accept a regular expression as input, and produce the layout of a recognizer for that expression. Regular expressions have seen wide application in computer science; among other tasks, they have been used to specify lexical analyzers for programming languages, controllers for sequential machines, filters for on-the-fly database search, patterns in image processing, and communication protocols. Regular expression recognizers therefore have wide application, and are an especially promising application area for specialized silicon compilers.

The contribution of this paper is a ''source-to-source'' transformation of regular expressions that allows more efficient recognizers to be constructed for them. The most straightforward circuits for constructing regular expression recognizers exhibit a problem: recognizers for some expressions contain superfluous latches due to interactions between the cells. Formation of these latches leads to recognizers that function incorrectly. Previous solutions to the problem of latch formation have used more complicated circuits to eliminate or reset the latches. These more complicated circuits increase the size of recognizers, and may make them run more slowly. The transformation introduced in this paper eliminates the expressions that cause latch formation, and so allows the simple, straightforward circuits to be used for all expressions. This allows specialized silicon compilers to produce efficient recognizers for regular languages.

2. Regular Expression Recognizers

This section gives a notation for regular expressions and describes how to compile them into circuits. Straightforward compilation of some regular expressions leads to a problem called *latch formation*, in which extraneous latches are formed within the compiled circuit. These latches lead to incorrect operation. This section indicates how the problem arises and mentions some techniques that have been used to combat it.

A regular expression describes a regular language over some alphabet Σ. A regular expression may represent the empty set ($\varnothing$), the empty string (ε), or any set of strings that can be built up by concatenation, union and repetition from ε and the single characters of Σ. A regular expression over Σ may include some characters that are not in Σ, such as operators and parentheses. Assuming that the characters in the set $\{\phi\ (\ )\ *\ +\}$ are not in Σ, the syntactically correct regular expressions over Σ can be defined inductively as follows.

- ϕ is a regular expression over Σ.
- If $a \in \Sigma$ then a is a regular expression over Σ.
- If α and β are regular expressions over Σ, then so are $\alpha;\beta$, $(\alpha + \beta)$, and $(\alpha)^*$.

The meaning of a regular expression can be defined inductively based on the form of the expression. The set of strings $L(\rho)$ represented by a regular expression ρ is:

- The empty set if ρ is ϕ.
- $\{a\}$ if ρ is a.
- $L(\alpha) \cup L(\beta)$ if ρ is $(\alpha + \beta)$.
- $\{\sigma_1\sigma_2$, where $\sigma_1 \in L(\alpha)$ and $\sigma_2 \in L(\beta)\}$ if ρ is $\alpha;\beta$.
- $\{\varepsilon\} \cup \{\sigma_1\sigma_2 \ldots \sigma_n$, where n is any positive integer and $\sigma_i \in L(\alpha)\}$ if ρ is $(\alpha)^*$. In this article, λ will be used as an abbreviation for ϕ^*. Thus, $L(\lambda) = \{\varepsilon\}$.

This article uses a regular expression to denote the set of strings it represents; for example, $abc \in (a;b + c)^*$.

Regular expressions describe patterns; very often one wants to build pattern recognizers for these patterns. A recognizer circuit for a regular expression accepts a stream of characters from the alphabet of the expression as input, and produces a stream of bits. The operation of a recognizer circuit occurs on discrete clock ticks, or *beats*. On each beat, the recognizer reads a character from the input and produces an output bit. The first bit precedes the string of characters, then the recognizer produces one bit after reading each character. Each bit indicates whether the stream of characters immediately preceding it is in the language of the regular expression. For example, if a recognizer for the expression $(a;b + c)^*$ were given the stream *ababc* it would produce the bit stream 001011.

Several researchers have described tree-based techniques for compiling regular expressions into recognizer circuits [1, 7, 9, 10, 15]. All of these techniques produce tree-structured circuits in which the nodes of the tree correspond to characters of the regular expression. Edges of the tree carry text and state information between nodes.

The circuits of Mukhopadhyay are typical examples of tree-based recognizer circuits. These circuits use four types of cells: one for each of the three operators (union, concatenation, and Kleene *) and one to recognize individual characters. Figure 2-1 shows the comparator for individual characters. On each beat, a character is input at the same time as the ENB signal. The RES signal is set to true for the following beat if and only if ENB is true and the text character matches the pattern character.

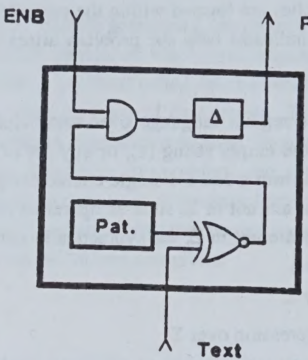


Figure 2-1: Comparator for expression-tree based recognizer

The three operator cells for the expression tree recognizers are shown in Figures 2-2, 2-3, and 2-4. These combine ENB and RES signals from their operands to produce signals for a larger expression. A recognizer built from these cells is a tree with the same structure as the expression tree of the regular expression, with operator cells in place of operators and comparators in place of letters of the alphabet. For example, to build a recognizer for A;B, a comparator for A is connected to the left port of the concatenation cell, and a comparator for B is connected to the right port. A recognizer constructed using these cells outputs RES on beat *i* if and only if some string in the language of the recognizer is input on beats *i*–*n* through *i*–1 and ENB_{i-n} is true.

There is a problem with the expression tree cells in Figures 2-1 through 2-4, which was first pointed out by Backhouse [4]. If a recognizer is constructed for an expression of the form E*, where the empty string is contained in E, the output result will be latched to true. This can be seen most easily in the recognizer for (*a**)*, shown in Figure 2-5. The OR gates in the two Kleene star cells are cross coupled so that they form a latch. If they are ever set to true, they can never be reset to false. Thus if ENB is true on beat 1, and the character *b* is input on that beat, the recognizer will erroneously output 1 on RES during beat 2. Using these cells then, correct recognizers can be constructed only for expressions in which the star operator is never applied to a subexpression containing the empty string. This problem is exhibited by most of the tree-based recognizers that have been discussed in the literature.

Several solutions to the latching problem have been proposed. Foster [9] uses a *clocked OR* gate in the Kleene closure cell in place of the OR gate. The clocked OR gate behaves like a normal OR gate, except that it briefly sets its output to 0 between beats. This clears any latches formed by a loop of OR gates, and it has been proven that recognizers built using this modified cell operate correctly for

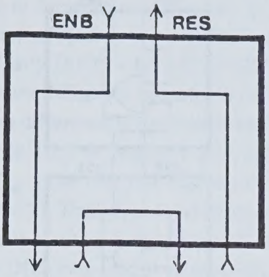


Figure 2-2: Concatenation cell

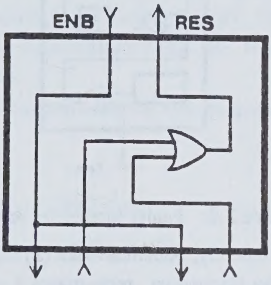


Figure 2-3: Union cell

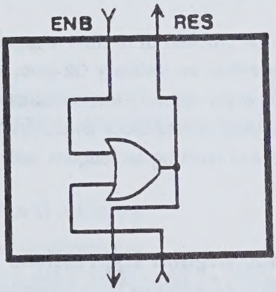


Figure 2-4: Kleene closure cell

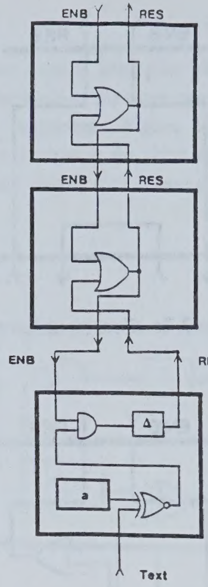


Figure 2-5: Faulty recognizer for (a*)*

all regular expressions [8]. More recently, Anantharaman [1] has designed a set of cells that can be interconnected to produce delay-insensitive recognizers. Among their other advantages, Anantharaman's cells avoid the latching problem by avoiding direct feedback between OR gates. The RES signals in these cells have more than two values; they take on a special value for matches of the empty string. The cells are designed so that the special value cannot act as an enable signal. In particular, a match of the empty string by a Kleene closure cell cannot re-enable the same cell.

While these techniques correct the problem of latched recognizers, they complicate the cells. The clocked OR gate is somewhat larger than an ordinary OR gate, and it requires connection to a clock that signals the ends of beats. The delay-insensitive circuits used in Anantharaman's cells are much larger and more complicated than their synchronous counterparts. In many cases in which a small, simple recognizer is desired, the anti-latching techniques add too much overhead to the prinitive cells.

3. A Conversion Theorem for Regular Expressions

This paper provides a new method for avoiding latch formation in tree-structured recognizers. The idea is to transform the regular expression for which a recognizer is to be built into a new expression that will not lead to latch formation. Latch formation occurs when the star operator is applied to an expression that contains the empty string; that is, any recognizer for an expression E*, where $\varepsilon \in E$, will have extraneous latches. If we could transform E into a new expression N(E) such that N(E)* = E* and $\varepsilon \notin N(E)$, we could avoid latch formation by building the recognizer for N(E)*. This recognizer would recognize the language E*, and would operate correctly using the simple cells of Figures 2-1 to 2-4.

The obvious way to transform E is to form the set difference E–λ. Although there is no way to build a cell for the set difference operator [8], E–λ is a regular language, and so can be represented by some regular expression D(E). Clearly D(E)* = E* and ε∉ D(E). Unfortunately, the expression D(E) can be much longer than E. The set difference operator is not used in regular expressions, and converting an expression containing set difference to one that does not can greatly increase the length of the expression. There is a sequence $\{E_n\}$ of regular expressions of increasing length such that $D(E_n)$ is about *n/4* times as long as E_n. (The first few expressions in the sequence are a*b*, (a* + b*)(c* + d*), (a*b* + c*d*)(e*f* + g*h*).) Thus there is no constant bound on the ratio of the length of D(E) to the length of E. Since the size of a tree-based recognizer is proportional to the length of its regular expression, transforming E to D(E) may lead to recognizers that are too large. We therefore need a different transformation of E.

The following theorem provides a transformation that is appropriate for constructing recognizers. It shows that any expression of the form E* can be transformed into a new expression N(E)* in which latches are not formed, and which is no longer than E*. This solves the problem of latch formation in tree-based recognizers for regular expressions. We first state without proof some easily verified facts about regular expressions.

Lemma : If P and Q are regular expressions the following statements hold.

a. P** = P*

b. (P* + Q*)* = (P + Q)*

c. If ε∈ P and ε∈ Q then (P;Q)* = (P + Q)*.

d. ε∈ (P + Q) if and only if either ε∈ P or ε∈ Q.

e. ε∈ (P;Q) if and only if ε∈ P and ε∈ Q.

These facts are used in the proof of the following theorem.

Theorem : For any regular expression R there is a regular expression N(R) such that

- N(R) does not contain the empty string,
- R* = (N(R))*
- N(R) is no longer than R.

Proof: We compute N(R) recursively, based on the expression tree of R. If ε∉ R then N(R) = R. (This grounds the recursion, since N(a) = a for any member of the alphabet.) Otherwise, there are three cases.

1. R = P*
2. R = P + Q, where either P or Q contains ε
3. R = P;Q, where both P and Q contain ε

In case 1, N(R) = N(P). In cases 2 and 3, N(R) = N(P) + N(Q). Clearly N(R) is no longer than R. We must now show that N(R) does not contain ε and that N(R)* = R*

We can show both of these facts by induction. First, we show that N(R) does not contain ε. If R has length 1, then it must be either a letter of the alphabet, or one of the expressions ϕ or λ. N(R) is then either ϕ or a letter of the alphabet, neither of which contains ε. Thus the base case of the induction holds. To prove the inductive step, suppose that for every expression P of length less than *i*, N(P) does not contain ε. Suppose R has length *i*. If R does not contain ε, neither does N(R). If R does contain ε, then N(R) is either N(P) or N(P) + N(Q), where P and Q have length less than *i*. In neither case does N(R) contain the empty string, since by the inductive hypothesis neither N(P) nor N(Q) can contain it.

Now we show that R* = (N(R))*. We need only consider cases in which R contains the empty string, since N(R) and R are identical otherwise. If R has length 1 and contains the empty string, it must be λ. To prove the base case, note that N(R) is then ϕ, and $(N(R))^* = R^* = \lambda$. For the inductive step, suppose that $R^* = (N(R))^*$ as long as the length of R is less than i. If R has length i and contains the empty string, then R must be one of the three forms noted above.

1. If $R = P^*$, then $N(R) = N(P)$ and $(N(R))^* = (N(P))^*$. By the inductive hypothesis, $(N(P))^* = P^*$. So $(N(R))^* = P^* = P^{**} = R^*$.
2. If $R = P + Q$, then $N(R) = N(P) + N(Q)$. So $(N(R))^* = (N(P) + N(Q))^* = ((N(P))^* + (N(Q))^*)^*$. By the inductive hypothesis, $(N(P))^* = P^*$ and $(N(Q))^* = Q^*$, so $(N(R))^* = (P^* + Q^*)^*$. But this is equal to $(P + Q)^*$, which is R^*.
3. If $R = P;Q$, then $N(R) = N(P) + N(Q)$. By the same reasoning as in case 2, $(N(R))^* = (P + Q)^*$. Now notice that, since we are assuming that R contains the empty string, both P and Q must also contain the empty string. Thus $R^* = (P;Q)^* = (P + Q)^*$. Therefore, $R^* = (N(R))^*$.

Some examples may clarify this theorem. Given a regular expression R, we can apply the theorem recursively to produce N(R). The reader may verify the following identities:

- $N(a^*) = a$
- $N(a^*;b^*;c^*;d^*) = a + b + c + d$
- $N((a^* + b^*);(c^* + d^*)) = a + b + c + d$

In these examples, a recognizer for $N(R)^*$ is smaller than a recognizer for R^*. Although this contraction in size may not always occur, Theorem 2 states that the recognizer for $N(R)^*$ will never be larger than the one for R^*.

4. Conclusion

This paper has described the problem of latch formation in regular expression recognizers, and has suggested a new solution. By transforming the regular expression R into N(R), not only may latches be avoided, but the corresponding recognizer may decrease in size. This source-to-source transformation seems to be a worthwhile addition to silicon compilers that produce regular language recognizers.

References

[1] Anantharaman, T. S.
A Delay Insensitive Regular Expression Recognizer.
1986.
Private Communication.

[2] Anantharaman, T. S., E. M. Clarke, M. J. Foster, and B. Mishra.
Compiling Path Expressions into VLSI Circuits.
In *Proceedings of the 12'th Symposium on Principles of Programming Languages*. ACM, January, 1985.
To appear in *Distributed Computing*.

[3] Anceau, F.
CAPRI: A Design Methodology and Silicon Compiler for VLSI Circuits.
In R. Bryant (editor), *Proceedings of the Third Caltech Conference on Very Large Scale Integration*, pages 15-32. Caltech, Pasadena, Ca., March, 1983.

[4] Backhouse, R. C.
Specification and Proof of a Regular Language Recognizer in Synchronous CCS.
Technical Report CSM-53, University of Essex, January, 1983.

[5] Balraj, T. S. and M. J. Foster.
Miss Manners: A Specialized Silicon Compiler for Synchronizers.
In *Proceedings of the Fourth MIT Conference on Advanced Research in VLSI.* MIT, April, 1986.

[6] Bergmann, N.
A Case Study of the F.I.R.S.T. Silicon Compiler.
In R. Bryant (editor), *Proceedings of the Third Caltech Conference on Very Large Scale Integration*, pages 413-430. Caltech, Pasadena, Ca., March, 1983.

[7] Floyd, R. W. and J. D. Ullman.
The Compilation of Regular Expressions into Integrated Circuits.
JACM 29(3):603-622, July, 1982.

[8] Foster, M. J.
Specialized Silicon Compilers for Language Recognition.
PhD thesis, Carnegie-Mellon University, 1984.

[9] Foster, M. J.
A Specialized Silicon Compiler and Programmable Chip for Language Recognition.
In N. G. Einspruch (editor), *VLSI Electronics: Microstructure Science.* Volume 14: *VLSI Design*, chapter 5, pages 139-196. Academic Press, Orlando, Florida, 1986.

[10] Foster, M. J. and H. T. Kung.
Recognize Regular Languages With Programmable Building Blocks.
Journal of Digital Systems 6(4):323-332, 1982.
A preliminary version of this paper appears in the VLSI-81 Proceedings, edited by John P. Gray.

[11] Hitchcock, C. Y. and D. E. Thomas.
A Method of Automatic Data Path Synthesis.
In *Proceedings of the 20'th Design Automation Conference.* IEEE, June, 1983.

[12] Karlin, A. R., H. W. Trickey, and J. D. Ullman.
Experience with a Regular Expression Compiler.
In *Proceedings of the International Conference on Computer Design*, pages 656-665. IEEE, October, 1983.

[13] Kowalski, T. J. and D. E. Thomas.
The VLSI Design Automation Assistant; Prototype System.
In *Proceedings of the 20'th Design Automation Conference.* IEEE, June, 1983.

[14] Lyon, R. F.
A Bit-Serial VLSI Architectural Methodology for Signal Processing.
In J. P. Gray (editor), *VLSI 81*, pages 131-140. University of Edinburgh, University of Edinburgh, August, 1981.

[15] Mukhopadhyay, A.
Hardware Algorithms for Nonnumeric Computation.
IEEE Transactions on Computers C-28(6):384-394, June, 1979.

[16] Savage, J. E.
The VLSI Compilation Techniques: PLA's; Weinberger Arrays; and SLAP, a New Silicon Layout Program.
Technical Report CS-82-24, Brown University, October, 1982.

[17] Southard, J. R.
MacPitts: An Approach to Silicon Compilation.
Computer 16(12):74-82, December, 1983.

ON THE POWER OF PROBABILISTIC ONE-TAPE TURING MACHINES

BALASUBRAMANIAN KALYANASUNDARAM [1] AND GEORG SCHNITGER [2]
Department of Computer Science,
The Pennsylvania State University.

Abstract : We have obtained a time bound of $\Omega(n^2/log^4 n)$ on the simulation of two-tape deterministic Turing machines by one-tape probabilistic Turing machines with a special one-way input tape. On the other hand we show the power of probabilism in this model by exhibiting a language that can be accepted in time O(n logn) by a one-tape probabilistic Turing machine but requires a time bound of $\Omega(n^2)$ for any one-tape nondeterministic Turing machines. These lower bounds are obtained with the help of arguments from Communication Complexity.

INTRODUCTION

Many on-line lower bounds [6], [4], [5], [3] have been proven recently showing that even stronger computational power can not make up for a weak geometrical storage structure. These results are complemented for instance in [7], [4], [3] where it is shown that bigger computational power pays off when the storage structure is unchanged.

We investigate one-tape Turing machines with a special one-way input tape (called ITM's from now on). We are especially interested in probabilistic ITM's. ITM's were considered in [4], [5] and [3]. Maass obtained an $\Omega(n^2)$ lower bound for deterministic ITM's simulating deterministic two-tape TM's. In Corollary 1 we prove that probabilistic ITM's still require $\Omega(n^2/log^4 n)$ steps for the simulation.

We use the geometric arguments developed by Maass. So far all the languages considered for ITM's are sparse in order to support a Kolmogorov-Complexity argument. But currently the methods for obtaining strong probabilistic bounds for sparse languages are too weak, see [1]. Therefore we introduce arguments from Communication Complexity. Corollary 1 follows from Theorem 1 where we obtain $\Omega(n^2)$ lower bounds for deterministic (nondeterministic, probabilistic) ITM's recognizing languages that are "hard" for deterministic (nondeterministic, probabilistic) information transfer.

---

[1] Supported by DN Initiatives N00014-80-C-0517

[2] Supported by NSF grant DCR84-07256 and Navy grant N000 14-80-C-0517

In Corollary 3, as another application of Theorem 1, we exhibit a language that can be recognized "fast" by a probabilistic ITM, but requires $\Omega(n^2)$ steps on a nondeterministic ITM.

For a fixed language $L \subset \{0,1\}^{2n}$ we define the languages D(L), ND(L) and P(L).

$D(L)$: Consider a triple (x,b,y) with |x|=2n, |y|=2n and |b|=1. (x,b,y) will always be rejected unless y has exactly n 0's. If b=0 then the triple belongs to $D(L)$ if and only if y specifies a number between 1 and 2n and the yth bit of x is 1. If b=1 then y splits x into two subsequences x_1 (specified by the 0's in y) and x_2 (specified by the 1's in y). The triple (x,b,w) belongs to $D(L)$ if and only if (x_1,x_2) belongs to L.

$ND(L)$: Consider a pair (x,y#z) such that |x|=2n, |y|=2nlog2n and |z|=2n. We interpret y as a sequence of 2n bit positions of x. Rearrange x according to y to obtain the string (x_1,x_2) where $|x_1| = |x_2| =$ n. The pair (x,y#z) belongs to the language $ND(L)$ if and only if (x_1,x_2) belongs to L and x=z.

$P(L)$: Consider a triple (x,b,y) where |x|=2n, |b|=1 and |y|=2nlog(2n). If b=1, then we again interpret y as a sequence of 2n bit positions of x and rearrange x according to y to obtain the string (x_1, x_2) where $|x_1| = |x_2| =$ n. The triple (x,1,y) belongs to the language $P(L)$ if and only if (x_1,x_2) belongs to L. The triple (x,0,y) belongs to $P(L)$ if and only if the number specified by y lies between 1 and 2n and the yth bit of x is 1.

We have to introduce some notions of Communication Complexity.

For a language $L \subset \{0,1\}^{2m}$ we define the 0/1 (2^m by 2^m) matrix $M(L)$ by $M(L)[x_1,x_2] = 1$ if and only if (x_1,x_2) belongs to L. We define the size of the information transfer problem to be m. The submatrix of $M(L)$ that we obtain after fixing arbitrarily f bit positions in x_1 and the corresponding bit positions in x_2 is called an *f-submatrix* of $M(L)$.

We say that L is (δ_1,δ_2)-*hard* deterministically (nondeterministically, probabilistically) if and only if for any $(\delta_1 m)$-*submatrix* X of $M(L)$ and any submatrix Y of X of size at least $2^{(1-\delta_2)(1-\delta_1)2m}$

deterministically : the deterministic communication complexity of the language of Y is at least $\delta_2 n$

nondeterministically : the nondeterministic communication complexity of the language of Y is at least $\delta_2 n$

probabilistically : Y will contain at least $\delta_2 | Y |$ 0-entries.

Theorem 1: Assume that for any δ_1 there is δ_2 so that L is (δ_1, δ_2)-hard deterministically (nondeterministically , probabilistically). Then any deterministic (nondeterministic , probabilistic) ITM that recognizes $D(L)$ ($ND(L)$, $P(L)$) requires at least $\Omega(n^2)$ time steps.

Corollary 1: Set $S(x_1,...,x_n; y_1,...,y_n) = \sum_{i=1}^{i=n} x_i \, y_i \ mod\, 2$. Then $P(S)$ can be recognized in $O(n \log^2 n)$ steps on a two-tape deterministic TM, but requires $\Omega(n^2)$ steps on any probabilistic ITM.

Proof: This follows directly from Theorem 1 and [2].

We also get immediately lower bounds for nondeterministic ITM's comparable to those of [4].

Corollary 2: Let S be the language of palindromes of length 2n. Then the language $ND(S)$ can be accepted by a deterministic two-tape TM in $O(n\log^2 n)$ steps, but it requires $\Omega(n^2)$ steps on any nondeterministic ITM.

Proof: The lower bound again follows directly from Theorem 1.

On the other hand for certain languages probabilism turns out to be even more powerful than nondeterminism. The following language M will exhibit the power of probabilism.

> Let $n=(2m)^2$. Consider a pair (x,y#z) such that |x|=n, |y|=4m logm and |z|=n. Imagine that x is divided into 2m consecutive blocks of size 2m. The first as well as the last 2m logm bits of y induce two perfect matchings M_0 and M_1 on the blocks of x. (Blocks 1,..,m are matched with blocks m+1,..,2m.) (If x is different from z or y does not induce perfect matchings then the pair will be rejected.) The pair (x,y#z) will belong to M if and only if for any two blocks belonging to M_0 the block contents coincide in the even bit positions and for any two blocks belonging to M_1 the block contents coincide in the odd bit positions.

Corollary 3: M can be recognized in O(n logn) steps on a probabilistic ITM. But any nondeterministic ITM requires $\Omega(n^2)$ steps.

Proof: The lower bound can be obtained along the lines of the proof for Theorem 1. For the upper bound we use ideas from [8].

Choose a prime p in the range $1 < p \leq m$ at random with all the primes in this range being equally likely. Assume that the work tape has six tracks.

We form groups of log m consecutive cells and call each group a column. We imagine the input tape to be divided into 2m consecutive blocks of size 2m each. For input x we have the blocks $x_{1,1}, x_{2,1}, \dots, x_{m,1}, x_{1,2}, x_{2,2}, \dots, x_{m,2}$. In track 1 and column i store $x_{i,1}$ (restricted to the even bit positions) mod p and in track 2 store $x_{i,1}$ (restricted to the odd bitpositions) mod p ($1 \le i \le m$). Also store $x_{i,2}$ (restricted to even positions) mod p in track 3 and finally $x_{i,2}$ (restricted to odd positions) mod p in track 4. All of this can be done in O(n logn) steps [8].

y consists of two queries y_1 and y_2. The query y_1 is divided into m consecutive blocks of size log m. In track 5 and column i+1 write the binary representation of i. In track 6 mark all the m columns *notseen* . Just before reading the m blocks of y_1 move the head of the work tape to the start of the first column. Assume that the log m bits in the first block of y_1 correspond to the binary representation of a number j. Move the work tape head to the start of the jth column. If column j is already *seen* , then reject and stop. Otherwise mark column j *seen* in track 6 and in track 7 write the binary representation of 0. Repeat this for all the m blocks of y_1, each time reading the column address from y_1. Especially for the ith block of y_1 write in track 7 the binary representation of i-1 in the column whose address is given in ith block of y_1. At the end mark all columns *not seen* .

We have described the algorithm for query y_1. We handle the query y_2 analogously.

Now with a straight forward deterministic algorithm we compare the residues mod p as demanded by the two matchings and reject appropriately.

Finally we have to read z. We use the terminology from above. We treat the blocks $z_{i,j}$ of z exactly as we treated the blocks of x (only now we use tracks 8,9,10 and 11 for the residues). It is now easy to compare the block residues of x with the block residues of z in O(n logn) steps.

The whole procedure takes only O(n logn) time steps. Additionally, with [8], we obtain constant error probability.

Proof of Theorem 1 for the probabilistic case

First we introduce Kolmogorov Complexity.

Definition 1:

(a) Let U be a universal Turing machine.

(b) Let x and y be words over {0,1}. Then $K(x \mid y)$ is defined to be the length of the shortest program p so that U on input (p, y) outputs x.

(c) Let S be a finite set of words. A element x of S is called random given y if and only if $K(x \mid y) \ge \log(|S|)$.

We call x an input and y a query. We have to consider a deterministic ITM with a special random tape. The random tape is filled with bits and the deterministic ITM consults this random tape and moves appropriately. Let r be the contents of the random tape. It suffices to show that there is a query y so that any deterministic ITM errs on at least $\Omega(2^{2^n})$ inputs. Fix a string r for the rest of the proof. We will show that for a query y that corresponds to a Kolmogorov- random matching (given r) any deterministic ITM errs on $\Omega(2^{2^n})$ inputs.

Finally we fix an ITM M and assume that it runs in νn^2 steps. It suffices to show that the error probability of M is at least constant.

Definition 2:

Let x be an input,p and q denote cells on the work tape.

1. Let CS(x,p) denote the crossing sequence for input x between work tape cells p and p+1 before reading the query y. Also let Read(x,p) be the set of input positions read when the work tape head crossed the boundary between cells p and p+1.

2a. Let $Left(x,p)$ ($Right(x,p)$) denote the set of input bit positions read for the first time while the work tape head is to the left(right) of the boundary of cells p and p+1. $LEFT(x,p)$($RIGHT(x,p)$) is the concatenation of those bits of x whose bit positions belong to $Left(x,p)$ ($Right(x,p)$).

b. Let $Middle(x,p,q)$ denote the set of input positions read for the first time while the work tape head is to the right of the boundary between cells p and p+1 or to the left of the boundary between cells q and q-1. Again, $MIDDLE(x,p,q)$ is the concatenation of those bits of x whose positions belong to $Middle(x,p,q)$.

3. We say that input x and cell p are compatible if and only if

$|Left(x,p)| \geq n/8$ and $|Right(x,p)| \geq n/8$.

4. Let Desert(x) be the set of cells p such that input x and p are compatible. Observe that Desert(x) is an interval.

5a. Let X and Y be subsets of 1,..,n . c and d are crossing sequences. Let Input(p,c,X;q,d,Y) be the set of all inputs x with

(1) c is the crossing sequence of x between position p and p-1 and d is the crossing sequence of x between positions q and q+1.

(2) X = Read(x,p-1) and Y = Read(x,q).

b. For a word m let Input(p,c,X;m;q,d,Y) denote the set of all elements x of Input(p,c,X;q,d,Y) with $Middle(x,p,q) = m$.

Case 1: There exist 2^{2n-1} inputs each of which is compatible with at least $\nu^{1/2}$n cells.

Definition 3: A set of pairs is said to have the *rectangle property* if and only if the pairs (x,y) and (u,v) belong to the set whenever the pairs (x,v) and (u,y) belong to the set.

Proposition 1: Let p and q be cells on the worktape, c and d are crossing sequences. X and Y are two subsets of {1,..,n}. Fix a word m. Then the set
{ ($LEFT(x,p)$,m,$RIGHT(x,p)$)| x ϵ Input(p,c,X;m;q,d,Y) } has the *rectangle property*.

Proof: Consider two elements x and y of Input(p,c,X;q,d,Y) with $Middle(x,p,q) = Middle(y,p,q) = m$. Define the input z to be identical to x on $Left(x,p)$ and $Middle(x,p,q)$ and to be identical to y on $Right(x,p)$. Now the ITM M, on input z, will also produce the crossing sequences c and d. Additionally we will have Read(z,p-1) = X and Read(z,q) = Y. This implies that z also belongs to Input(p,c,X;q,d,Y) and we have proved the proposition.

Fix cells p and q,crossing sequences c and d and subsets X and Y of {1,..,n}. Observe that there are sets L(p,c,X) and R(q,d,Y) so that for each x in Input(p,c,X;q,d,Y)
L(p,c,X) = Left(x,p-1) and R(q,d,Y) = Right(x,q).

Definition 4: A query y defines a matching on the 2n bit positions. We say that
y induces an information transfer problem of size m on Input(p,c,X;q,d,Y) if and only if
if L is the set of those bit positions in L(p,c,X) that are matched by y with bit positions in R(q,d,Y), then |L| = m.

Lemma 2: Let y be a query that corresponds to a Kolmogorov-random matching.
Assume that $\min(\,|L(p,c,X)|\,,\,|R(q,d,Y)|\,) = \rho$n. Also assume |X|,|Y| $\leq \delta n$ and |c|,|d| $\leq \delta n$. Then y induces an information transfer problem of size $(\rho^2 - f(\rho,\delta))$n on Input(p,c,X;q,d,Y). Here f(u,v) tends towards 0 with v approaching 0.

Proof: By a conventional Kolmogorov Complexity argument.

Lemma 3: Let y be a query that corresponds to a Kolmogorov-random matching. Then the ITM M will err on g(ν)2^{2n} inputs.
Here g(x) is a non-negative function which is not identically zero on [0,1].

Proof : Consider the set Input($p,c,X;q,d,Y$). We say that X and Y "guarantee" the μ-condition ($1/8 \leq \mu < 1/2$) if and only if for all x ϵ Input($p,c,X;q,d,Y$):

at most μn input bits are read for the first time to the left of p-1 (to the right of q+1) and more

than μn input bits are read for the first time to the left of p (to the right of q).

Let us call the vector $(p,c,X;q,d,Y)$ *good* for μ iff

(a) $|\text{Input}(p, c, X; q, d, Y)| \geq 2^{(1-O(\nu^{1/2}))2n}$ and $|c|, |d| \leq 2\nu^{1/2}n$

(b) X and Y "guarantee" the μ-condition.

Now for sufficiently small ν the assumption of *Case* 1 implies, that we will find μ_0 and a set I of inputs so that

$|I| = \Omega(2^{2n})$ and each element of I belongs to the set induced by a *good* vector for μ_0.

First, as a consequence of Proposition 1, we observe that for any word m the set Input(p, c, X;m;q, d, Y) has the *rectangle property*. Also observe that two different *good* vectors induce disjoint sets of inputs. Now consider one *good* vector.

With Lemma 2 a query y will produce an information transfer problem of size at least s $= (\mu_0{}^2 - f(\mu_0, 2\nu^{1/2}))n$. We will find one assignment of the unmatched bit positions so that at least $|\text{Input}(p, c, X; q, d, Y)|/2^{(1-s/n)2n}$ inputs will belong to Input(p, c, X;m;q, d, Y) for some m. This set forms a submatrix *Sub* of a (1-s/n)n-submatrix of L. Since *Sub* will be of size at least $2^{(1-O(\nu^{1/2})-(1-s/n))2n} = 2^{(s/n-O(\nu^{1/2}))2n}$, we will get constant error probability. This follows from the property of L and the small length of the crossing sequence.

Remove the entries covered by *Sub* from I. Repeat assigning values to the unmatched bit positions until $\frac{1}{2}\ |I|$ elements have been removed from I. But this proves the Lemma.

Case 2: Case 1 is false.

Let J be the set of those 2^{2n-1} inputs with less than $\nu^{1/2}n$ compatible points on the work-tape. So these inputs have deserts of size less than $\nu^{1/2}n$. Observe that at most $n/8$ bit positions are read for the first time when the head is to the left (right) of the desert.

Fact 5: $\binom{2n}{\delta n} \leq (e\,/2\delta)^{\delta n}$.

Lemma 6: Under the assumption of Case 2, the ITM M will err on $\Omega(2^{2n})$ inputs.

Proof : There will be one interval [a,b] (with b-a $\leq \nu^{1/2}n$) which will occur as desert for at least $j = |J|/n^4$ inputs. (We assume here a running time of less than n^2).

We apply Fact 5 with $\delta = n/4$ and obtain

$$\binom{2n}{n/4} \leq (8e)^{n/4} \leq 2^{5n/4}.$$

Therefore we will find a set I of $7n/4$ bit positions and a subset J_1 of J of size at least $j_1 = j/2^{5n/4}$ so that each input in J_1 reads all bit positions in I for the first time in the desert [a,b].

Now extend [a,b] to the left until a cell A is reached so that at least j_1/n^2 elements of J_1 have crossing sequences of length at most αn (with $\alpha = 32\nu$). Observe that $a - A \leq \nu/\alpha n$ where we

assume that the running time is νn^2. Next extend [A,b] to the right until a cell B is reached such that the crossing sequence is less than αn . Again we will find $B-b \leq \nu/\alpha n$.
We are therefore guaranteed to obtain a subset J_2 of J_1 of size at least $j_2 \geq j_1/n^4$ so that all elements of J_2 have crossing sequences of length at most αn at A and B.
Now consider a randomly chosen input x belonging to J_2. We claim that the ITM M makes mistakes for $\Omega(n)$ queries that extract a bit of the input. We sketch the argument. We observe that the number of bits read (= 7n/4) inside the desert for the first time by far exceeds the size of the desert
($\leq (2\nu/\alpha+\nu^{1/2})n < n/4$ for sufficiently small ν). Also the crossing sequences at A and B do not allow to communicate all bits to the outside (its length is at 8ν). Therefore M can't reconstruct x from the contents of the work tape.
(the Kolmogorov Complexity of x restricted to the desert is at least
(7n/4 - 5n/4 - o(n) = n/2 - o(n).)
This implies a constant proportion of mistakes in J_2.
Now we remove J_2 from J and repeat the above procedure until we have dealt with $J/2$ elements. So we shown constant error ratio for J and this completes the argument.

Bibliography

[1] L. Babai, P. Frankl and J. Simon: BPP and the Polynomial Time Hierarchy in Communication Complexity, TR 86-003 Feb '86, Univ. of Chicago.

[2] B. Chor and O. Goldreich: Unbiased Bits from Weak Sources of Randomness, Proc. 26th IEEE FOCS (1985).

[3] Z. Galil, R. Kannan and E. Szemeredi: On Nontrivial Separators for k- Page Graphs and Simulations by Nondeterministic One-Tape Turing Machines, Proc. 18th ACM STOC (1986).

[4] W. Maass: Quadratic Lower Bounds for Deterministic and Nondeterministic One-Tape Turing Machines, Proc. 16th ACM STOC (1984), pp. 401-408.

[5] M. Li:Lower Bounds by Kolmogorov Complexity, (1985) Proc. 12th Colloq. on Automata, Languages and Programming, pp. 383-393.

[6] R. Paturi and J.Simon: Lower Bounds on the Time of Probabilistic On-Line Simulations, Proc. 24th IEEE FOCS (1983), pp. 343-350.

[7] W. Paul, N. Pippenger, E. Szemeredi, W.T. Trotter, On Determinism versus Nondeterminism, Proc. 24th IEEE FOCS (1983), pp. 429-438..

[8] N. Pippenger: Probabilistic simulations, Proc. 14th ACM STOC (1982), pp. 17-24.

PARALLEL ALGORITHMS FOR SOME FUNCTIONS OF TWO CONVEX POLYGONS[†]

MIKHAIL J. ATALLAH
MICHAEL T. GOODRICH
Dept. of Computer Science
Purdue University
West Lafayette, IN 47907

Abstract

Let P and Q be two convex, n-vertex polygons. We consider the problem of computing, in parallel, some functions of P and Q when P and Q are disjoint. The model of parallel computation we consider is the CREW-PRAM, i.e. it is the synchronous shared-memory model where concurrent reads are allowed but no two processors can simultaneously attempt to write in the same memory location (even if they are trying to write the same thing). We show that a CREW-PRAM having $n^{1/k}$ processors can compute the following functions in $O(k^{1+\varepsilon})$ time: (i) The common tangents between P and Q, (ii) The distance between P and Q (and hence a straight line separating them). The positive constant ε can be made arbitrarily close to zero. Even with a linear number of processors, it was not previously known how to achieve constant time performance for computing these functions.

1. Introduction

Let P and Q be two convex, n-vertex polygons. We consider the problem of computing, in parallel, the following functions of P and Q when P and Q are disjoint : (i) The common tangents between P and Q, (ii) The shortest distance between P and Q (and hence a line separating them). Throughout the paper, the model of parallel computation we use is the CREW-PRAM, i.e. it is the synchronous shared-memory model where concurrent reads are allowed but no two processors can simultaneously attempt to write in the same memory location (even if they are trying to write the same thing). Let c and d be any integers of our choice, and let $k=c^d$. We show that a CREW-PRAM having $n^{1/k}$ processors can compute the above-mentioned functions in time $O(k^{1+\varepsilon(c)})$ where $\lim_{c \to \infty} \varepsilon(c) = 0$; hence c can be chosen to be a constant that is large enough to make $\varepsilon(c)$ very close to zero. Our algorithms are not straightforward parallel generalizations of the known sequential algorithms [4,5] for these problems, but are indeed very different.

Setting $k=1$ in our common tangents result immediately implies an optimal $O(\log n)$ time, n processor parallel convex hull algorithm that is simpler than the ones recently given in [1,2]. The parallel convex hull algorithms given in [1,2] avoid the cleaner approach of recursively solving two subproblems of size $n/2$ each [3,6,7], because it was not known then how to find the common tangents between the two subsolutions in constant time and with n processors. Instead, these previous parallel convex hull algorithms partition the input points into $\sqrt{n}$ sets of size $\sqrt{n}$ each and, although asymptotically optimal, they are less natural than the standard solution [3,6,7] whose efficient parallel implementation is made possible by this paper. Essentially the same technique that we use for establishing our common tangents result is used to design a parallel algorithm for computing the shortest distance between P and Q.

The paper is organized as follows. Section 2 gives the algorithm for computing the common tangents between P and Q. Section 3 gives a similar result for computing the (shortest) distance between P and Q. Section 4 concludes.

---

† This research was supported by the Office of Naval Research under Grants N00014-84-K-0502 and N00014-86-K-0689, and the National Science Foundation under Grant DCR-8451393, with matching funds from AT&T.

2. Finding Common Tangents

Let $P=(p_1,\cdots,p_n)$ and $Q=(q_1,\cdots,q_n)$ be two disjoint convex polygons, where the p_i's (resp. q_i's) are given in clockwise cyclic order. For convenience, we assume that no three successive vertices of either polygon are colinear. Let c and d be any integers of our choice, $c=O(1)$. Let $k=c^d$. Our aim is to show that a CREW-PRAM with $n^{1/k}$ processors can compute the two common tangents between P and Q (P and Q are on the same side of a common tangent) in time $O(k^{1+\varepsilon(c)})$, where $\lim_{c\rightarrow\infty}\varepsilon(c)=0$. By choosing c to be a large enough constant, we can make $\varepsilon(c)$ arbitrarily close to zero. As already mentioned, even the case $k=1$ of this result was previously an open question.

Since P and Q are disjoint, they are separable by a straight line. Such a separating line is not given as part of the input. However, a by-product of the algorithm we give in Section 3 is that an $n^{1/k}$ processor CREW-PRAM can, in $O(k^{1+\varepsilon(c)})$ time, find a straight line separating P and Q. For the rest of this section, we assume that such a separating line (call it L) has already been found. Without loss of generality, we assume that L is vertical, that P is to its left and Q is to its right. We focus on the problem of computing the upper common tangent (that of computing the lower one being symmetrical), and we henceforth use P and Q to denote the upper portions of the two input polygons. For notational convenience, we continue to assume that P and Q are n-gons, i.e. that $|P|=|Q|=n$. See Figure 1.

If we had n^2 processors available, then it would be trivial to find the desired common tangent in constant time (the detailed specification of such a brute force algorithm is easy and is omitted). In view of this last remark, one may be tempted to give the following straightforward constant time, n processor "solution" (**which doesn't work**) :

(i) Consider two evenly spaced $\sqrt{n}$-subsequences of the vertices of P and Q, obtaining the two $\sqrt{n}$-gons $P'=(p_{\sqrt{n}},p_{2\sqrt{n}},\cdots,p_n)$ and $Q'=(q_{\sqrt{n}},q_{2\sqrt{n}},\cdots,q_n)$. Use the above-mentioned brute force approach to find the common tangent to P' and Q' in constant time. Say it is the line joining $p_{i\sqrt{n}}\in P'$ to $q_{j\sqrt{n}}\in Q'$.

(ii) The vertices of P' (resp. Q') divide P (resp. Q) into $\sqrt{n}$ portions, call them $P_1,\cdots,P_{\sqrt{n}}$ (resp. $Q_1,\cdots,Q_{\sqrt{n}}$). Use the brute force algorithm between the $2\sqrt{n}$ points in $P_i\bigcup P_{i+1}$ and the $2\sqrt{n}$ points in $Q_j\bigcup Q_{j+1}$ (i.e. between the portions of P adjacent to $p_{i\sqrt{n}}$ and the portions of Q adjacent to $q_{j\sqrt{n}}$).

The reason the above approach fails is that the "locality" property needed for Step (ii) need not hold: Indeed, the portion of P (resp. Q) containing the left (resp. right) point of tangency might be quite far from $p_{i\sqrt{n}}$ (resp. $q_{j\sqrt{n}}$). (We leave it to the reader to find an example of how this might happen.) The correct solution to the common tangent problem makes a more judicious use of the basic idea of the above (erroneous) steps (i)-(ii). It also makes use of the next two (easy) propositions.

Proposition 1. Let p be a point external to Q. Then the upper tangent to Q passing through p can be computed in time $O(k)$ by an $n^{1/k}$ processor CREW-PRAM, where k is any integer of our choice.

Proof. Let $t=n^{1-1/k}$. Let Q' consist of every t-th vertex of Q, i.e. $Q'=(q_t,q_{2t},\cdots,q_n)$. Since Q' has $n^{1/k}$ vertices and we have $n^{1/k}$ processors, it is trivial to find in constant time the upper tangent to Q' passing through p, say this tangent touches Q' at q_{it}. Let q_j be the vertex of Q at which the desired tangent touches Q. Test whether q_j is to the left of q_{it}, to the right of q_{it}, or at q_{it} (this test trivially takes constant time with one processor). If $q_j=q_{it}$ then we're done, so suppose (without loss of generality) that the test reveals that q_j is to the left of q_{it}, i.e. $j<it$ (the case $it<j$ is symmetrical). Then it is not hard to prove that we have $(i-1)t\leq j$ (we leave the proof to the reader). Therefore it suffices to find the upper tangent to polygon $(q_{it-t},q_{it-t+1},\cdots,q_{it-1})$ passing through p. Thus, by doing a constant amount of work, we have reduced the polygon size by a factor of $n^{1/k}$. Doing this at most k times finds the desired point of tangency. □

Proposition 2. Let p be a vertex of P and let p_u be a vertex of P at which the common tangent between P and Q touches P. Then for any integer k of our choice, an $n^{1/k}$ processor CREW-PRAM can, in $O(k)$ time, determine whether p_u is to the left of p, to the right of p, or at p.

Proof. Use Proposition 1 to find the tangent to Q passing through point p, let T be this tangent. If T is tangent to P then $p_u=p$. Otherwise, let γ be the point just to the left of p on P. It is obvious that p_u is to the left of p on P if and only if γ is above line T. □

The following preliminary algorithm shows that, for any integer c of our choice, an $n^{1/c}$ processor CREW-PRAM can find the common tangent to P and Q in $O(c^2)$ time.

Preliminary algorithm A for finding upper common tangent:

Input. The upper portions P and Q of two disjoint convex polygons separated by a vertical line L. Both $P=(p_1, \cdots, p_n)$ and $Q=(q_1, \cdots, q_n)$ are monotone in the x direction, i.e. the x component of p_i (resp. q_j) is smaller than that of p_{i+1} (resp. q_{j+1}). See Figure 1. *Note:* The assumption that we are already given L is not really needed, since Section 3 shows how to find such a line L.

Output. The upper common tangent to P and Q.

Step 0. Set $\hat{P}:=P$, $\hat{Q}:=Q$, $s:=n^{1/2c}$.

Step 1. Repeat the following steps 2-6 until either $\hat{P}$ is a single point or $\hat{Q}$ is a single point. Without loss of generality, assume that it is $\hat{P}$ that ends up becoming a single point (call it p_u). Use Proposition 1 to find, in $O(c)$ time, the tangent to $\hat{Q}$ passing through p_u, and output the tangent thus found (this is the desired tangent between P and Q).

Step 2. Let $P'=(a_1, \cdots, a_s)$ be the polygon obtained by considering every $(|\hat{P}|/s)$-th vertex of $\hat{P}$, i.e. the s vertices of P' divide $\hat{P}$ into s equal portions. Call these portions $A_1, \cdots, A_s$, so that a_i is adjacent in $\hat{P}$ to portions A_i and A_{i+1}. By definition, a_i belongs to A_i but not to A_{i+1}. Let $Q'=(b_1, \cdots, b_s)$ be analogously defined for $\hat{Q}$, and let the resulting portions of $\hat{Q}$ be $B_1, \cdots, B_s$. Use the already mentioned brute force method for finding the common tangent between P' and Q' (this is possible and takes constant time because we have s^2 processors). Say the tangent thus found joins $a_i \in P'$ to $b_j \in Q'$. (See Figure 2.)

Step 3. Test whether the common tangent to $\hat{P}$ and $\hat{Q}$ touches $\hat{P}$ in A_i. (This is done in $O(c)$ time by using Proposition 2 twice, once at vertex p_{i-1} and once at vertex p_i.) If the answer is "yes" then do $\hat{P}:=A_i$, otherwise $\hat{P}$ remains unchanged.

Implementation Note. The assignment $\hat{P}:=A_i$ is done in constant time simply by remembering the new first and last vertex of $\hat{P}$.

Step 4. Test whether the common tangent to $\hat{P}$ and $\hat{Q}$ touches $\hat{P}$ in A_{i+1}. If it does then do $\hat{P}:=A_{i+1}$, otherwise $\hat{P}$ remains unchanged.

Step 5. Test whether the common tangent to $\hat{P}$ and $\hat{Q}$ touches $\hat{Q}$ in B_j. If it does then do $\hat{Q}:=B_j$, otherwise $\hat{Q}$ remains unchanged.

Step 6. Test whether the common tangent to $\hat{P}$ and $\hat{Q}$ touches $\hat{P}$ in B_{j+1}. If it does then do $\hat{Q}:=B_{j+1}$, otherwise $\hat{Q}$ remains unchanged.

(End of algorithm).

Note that the algorithm maintains the property that the tangent between P and Q is the same as the tangent between $\hat{P}$ and $\hat{Q}$. Thus the algorithm is correct.

Since every usage of Proposition 2 takes $O(c)$ time, the time complexity of the algorithm is equal to c multiplied by the number of times that steps 2-6 get executed. We now bound the number of times steps 2-6 are executed.

Lemma 1. Let a_i, b_j, $\hat{P}$, $\hat{Q}$, P', and Q' be as defined in Step 2 of algorithm A. Also, let $p_u q_v$ be the common tangent to $\hat{P}$ and $\hat{Q}$ ($p_u \in \hat{P}$, $q_v \in \hat{Q}$). Then at least one of the following statements (a), (b), (c), or (d) is true:

(a) $p_u \in A_i$;

(b) $p_u \in A_{i+1}$;

(c) $q_v \in B_j$;

(d) $q_v \in B_{j+1}$.

Proof. If $p_u=a_i$ or $q_v=b_j$ then the lemma holds, so suppose that $p_u \neq a_i$ and $q_v \neq b_j$. By its definition, the line $p_u q_v$ is above both a_i and b_j. Therefore at least one of p_u or q_v is above the straight line $a_i b_j$. So suppose that p_u is above the line $a_i b_j$. We prove that (a) or (b) must hold by a case analysis.

Case 1: In $\hat{P}$, p_u is to the left of a_i. Then we claim that $p_u \in A_i$ (and hence (a) holds). Suppose to the contrary that $p_u \in A_w$ where $w<i$. By the definition of a_i and b_j, the vertex $a_w \in P'$ must lie on or below the straight line $a_i b_j$. The three vertices p_u, a_w, a_i occur in that order on $\hat{P}$ (see Figure 3). Consider the positions of these three vertices relative to the line $a_i b_j$: The first vertex is (by

hypothesis) above that line, the second is (as we have just argued) on or below it, and the third is (by definition) on it. This contradicts the convexity of $\hat{P}$. Thus, (a) holds.

Case 2: In $\hat{P}$, p_u is to the right of a_i. An argument similar to that for Case 1 shows that $p_u \in A_{i+1}$; hence, (b) holds.

If q_v is above line $a_i b_j$, then an argument very similar to that above shows that one of (c) or (d) must hold. □

Corollary 1. Steps 2-6 of algorithm A are executed a total of at most $4c-1$ times.

Proof. Lemma 1 implies that, every time we execute steps 2-6, at least one of the statements $\hat{P}:=A_i$, $\hat{P}:=A_{i+1}$, $\hat{Q}:=B_j$, $\hat{Q}:=B_{j+1}$ is executed. This implies that at least one of $\hat{P}$ or $\hat{Q}$ decreases in size by a factor of $s=n^{1/2c}$, thus proving the corollary. □

We have thus established the following

Theorem 1. Algorithm A correctly computes the upper common tangent to P and Q. With $n^{1/c}$ processors, it runs in time $O(c^2)$.

Corollary 2. With n processors, the upper common tangent to P and Q can be computed in constant time.

Let $\mathbf{B_0}$ be the algorithm corresponding to Corollary 2, i.e. $\mathbf{B_0}$ runs in $O(1)$ time with n processors. Now, we define a sequence of algorithms $\mathbf{B_1}, \mathbf{B_2}, \cdots$ such that $\mathbf{B_d}$ uses n^{1/c^d} processors, and is defined as follows: $\mathbf{B_d}$ reads exactly like $\mathbf{A}$ except that

(i) In $\mathbf{B_d}$, Step 0 sets s equal to $n^{1/c}$ (instead of $n^{1/2c}$ in A).

(ii) In Step 2, whereas $\mathbf{A}$ uses the brute force method, $\mathbf{B_d}$ uses $\mathbf{B_{d-1}}$ (we can do this even though there are only n^{1/c^d} processors available, because $\mathbf{B_{d-1}}$ is being used on a subproblem of size only $n^{1/c}$).

(iii) Every usage of Proposition 1 or Proposition 2 now costs $O(c^d)$ time because we have only n^{1/c^d} processors.

Obviously, Lemma 1 still holds for $\mathbf{B_d}$ just as it did for $\mathbf{A}$. Thus, every time steps 2-6 are executed in $\mathbf{B_d}$, at least one of $\hat{P}$ or $\hat{Q}$ decreases in size by a factor of $s=n^{1/c}$. This implies that steps 2-6 in $\mathbf{B_d}$ are executed at most $2c-1$ times. If we let T_d be the time complexity of $\mathbf{B_d}$, then we have

$T_0 = c_1$, and

$$T_d = (2c-1).(T_{d-1}+c_2.c^d) + c_3.c^d$$

where c_1, c_2 and c_3 are constants. This has solution $T_d = O(c^2(c-1)^{-1}(2c-1)^d)$. Choosing c to be a constant and using $k=c^d$ gives

$T_d = O(k^{1+\varepsilon(c)})$ where $\varepsilon(c) = \log_c(c^{-1}(2c-1))$. This establishes the following:

Theorem 2. Let P and Q be two disjoint convex n-gons. Let $k=c^d$ where c and d are any integers, $c=O(1)$. A CREW-PRAM having $n^{1/k}$ processors can compute the common tangents between P and Q in $O(k^{1+\varepsilon(c)})$ time, where $\lim_{c \to \infty} \varepsilon(c) = 0$.

3. Computing The Distance

The input consists of the two disjoint convex polygons $P=(p_1, \cdots, p_n)$ and $Q=(q_1, \cdots, q_n)$, where the p_i's (resp. q_i's) are given in clockwise cyclic order and no three successive vertices of either polygon are colinear. We are interested in computing, in parallel, the shortest distance between P and Q. This distance is formally defined as follows:

$$d(P,Q) = \min_{u \in P, w \in Q} d(u,w)$$

where $d(u,w)$ denotes the Euclidean distance between points u and w, and the notation "$u \in P$" means that u is a point on the boundary of P (not necessarily a vertex of P). Our algorithm actually returns a pair of points u,w such that $d(P,Q)=d(u,w)$. Of course, once we have these points u,w, any perpendicular to the straight line segment joining u and w is a line separating P from Q. Therefore our algorithm for the closest distance immediately gives us the separating line L needed in Section 2.

In order to simplify the exposition, we assume that the desired points u,w are unique. Our algorithm can easily be modified for the general case, e.g. by adopting a suitable convention for returning a unique u,w pair in case $d(P,Q)$ is the distance between two parallel segments of (respectively) P and Q (in that case there is an infinite number of choices for u,w, and this is the only case where u and w are not unique).

Let p be a point (not necessarily a vertex) on the boundary of P, and define q similarly for Q. Let T_p (resp. T_q) be the line perpendicular to the segment pq at point p (resp. q). It is quite trivial to see that $d(P,Q)=d(p,q)$ if and only if (i) T_p and T_q are tangent to (respectively) P and Q, and (ii) P and Q are on opposite sides of the region between T_p and T_q (i.e., this region separates them). This simple observation immediately implies that, with n^2 processors and in constant time, it is possible to compute the closest distance between P and Q and a pair of points achieving it (the detailed specification of this brute force procedure is left to the reader). The algorithm we shall give uses these simple observations. It also makes use of the next two (easy) propositions.

Proposition 3. Let p be a point external to Q. Then the point $q \in Q$ such that $d(p,q)=d(p,Q)$ can be computed in time $O(k)$ by an $n^{1/k}$ processor CREW-PRAM, where k is any integer of our choice.

Proof. Let $t=n^{1-1/k}$. Let Q' consist of every t-th vertex of Q, i.e. $Q'=(q_t, q_{2t}, \cdots, q_n)$. Since Q' has $n^{1/k}$ vertices and we have $n^{1/k}$ processors, it is trivial to find in constant time the point $q' \in Q'$ such that $d(p,q')=d(p,Q')$. (Note that q' need not be a vertex of Q'.) If the perpendicular to line pq' at point q' is tangent to Q, then we can stop and declare point q' as the desired point q. Otherwise let α (resp. β) be the vertex of Q' that immediately precedes (resp. follows) point q' when the boundary of Q' is traced in a clockwise manner (see Figure 4). Note that in Q, there are $2t+1$ vertices between α and β (inclusive) if q' is a vertex, otherwise there are $t+1$ vertices between α and β (where the word "between" refers to the circular ordering $q_1, \cdots, q_n$). We leave it to the reader to prove that, in Q, the desired point q occurs between α and β (inclusive). Let γ be the median of the (at most $2t+1$) vertices between α and β (inclusive): Test whether the desired point q is at γ, between α and γ, or between γ and β (this test trivially takes constant time with one processor). If $q=\gamma$ then we're done, so suppose (without loss of generality) that the test reveals that q is between α and γ. Hence we can focus our search for q to the section of Q between α and γ (excluding γ), which contains at most t vertices. Therefore by doing a constant amount of work, we have reduced the polygon size by a factor of at least $n^{1/k}$. Doing this at most k times finds the desired point q. □

Proposition 4. Let p_i and p_j be vertices of P, $i<j$, and let p_u be the vertex of P such that $d(p_u,Q)=d(P,Q)$. Then for any integer k of our choice, an $n^{1/k}$ processor CREW-PRAM can, in $O(k)$ time, locate where p_u occurs with respect to p_i and p_j in the sequence $p_1, p_2, \cdots, p_n$ (i.e. it can determine whether $u=i$, $u=j$, $i<u<j$, $u<i$ or $j<u$.)

Proof. Use Proposition 3 to test the sequence $d(p_1,Q), d(p_2,Q), \cdots, d(p_n,Q)$ in the vicinity of points p_i and p_j. More precisely, Proposition 3 is used to determine whether this sequence is increasing, decreasing, has a peak, or a valley, at points p_i and p_j. The sequence is *increasing at* p_i if $d(p_{i-1},Q) \le d(p_i,Q) \le d(p_{i+1},Q)$, it is *decreasing* if $d(p_{i-1},Q) \ge d(p_i,Q) \ge d(p_{i+1},Q)$, it has a *valley* if $d(p_{i-1},Q) \ge d(p_i,Q)$ and $d(p_i,Q) \le d(p_{i+1},Q)$, and it has a *peak* if $d(p_{i-1},Q) \le d(p_i,Q)$ and $d(p_i,Q) \ge d(p_{i+1},Q)$. It is trivial to see that this information is enough to determine precisely where p_u occurs with respect to p_i and p_j in the sequence $p_1, p_2, \cdots, p_n$. □

The following preliminary algorithm shows that, for any integer c of our choice, an $n^{1/c}$ processor CREW-PRAM can find, in $O(c^2)$ time, the points $u \in P$ and $w \in Q$ such that $d(u,w)=d(P,Q)$.

Preliminary Algorithm D for computing distance:

Input. Two disjoint convex polygons $P=(p_1, \cdots, p_n)$ and $Q=(q_1, \cdots, q_n)$. The p_i's (resp. q_j's) are given in clockwise cyclic order.

Output. Points u,w such that $d(u,w)=d(P,Q)$.

Step 0. Set $\hat{P}:=P$, $\hat{Q}:=Q$, $s:=n^{1/2c}$.

Step 1. Repeat the following steps 2-4 until either $\hat{P}$ is a single point or $\hat{Q}$ is a single point. Without loss of generality, assume it is $\hat{P}$ that ends up becoming a single point (call it x): Use Proposition 3 to find, in $O(c)$ time, the point $y \in Q$ such that $d(x,y)=d(x,Q)$. Output the points x and y (these are the desired points u,w).

Step 2. Let $P'=(a_1, \cdots, a_s)$ be the polygon obtained by considering every $(|\hat{P}|/s)$-th vertex of $\hat{P}$, i.e. the s vertices of P' divide $\hat{P}$ into s equal portions. Call these portions $A_1, \cdots, A_s$, so that a_i is adjacent in $\hat{P}$ to portions A_i and A_{i+1}. By definition, a_i belongs to A_i but not to A_{i+1}. Let $Q'=(b_1, \cdots, b_s)$ be analogously defined for $\hat{Q}$, and let the resulting portions of $\hat{Q}$ be $B_1, \cdots, B_s$.

Use the already mentioned brute force method for finding the points $a \in P'$ and $b \in Q'$ such that $d(a,b)=d(P',Q')$. Since we have s^2 processors, this takes constant time.

Let α_P (resp. β_P) be the vertex of P' that immediately precedes (resp. follows) a on the boundary of P'. If a is a vertex of P' then α_P and β_P are (respectively) its predecessor and successor vertices on P', and hence there are $2|\hat{P}|/s+1$ vertices of $\hat{P}$ between α_P and β_P (inclusive). If a is not a vertex of P' then α_P and β_P are consecutive vertices of P', point a is on the segment of P' that joins α_P to β_P, and there are $|\hat{P}|/s+1$ vertices of $\hat{P}$ between α_P and β_P (inclusive). Let γ_P be the median of the (at most $2|\hat{P}|/s+1$) vertices of $\hat{P}$ that are between α_P and β_P (inclusive). (Note that, if a is a vertex of P', then $\gamma_P=a$.) We use $P^{\alpha\beta}$ to denote the portion of P that is between α_P and β_P (excluding α_P and β_P). $P^{\alpha\gamma}$ and $P^{\gamma\beta}$ are analogously defined.

Let α_Q, β_Q, γ_Q, $Q^{\alpha\beta}$, $Q^{\alpha\gamma}$ and $Q^{\gamma\beta}$ be similarly defined for b, Q' and $\hat{Q}$. (See Figure 5.)

Step 3. Use Proposition 4 to detect whether $u=\alpha_P$, $u=\beta_P$, $u=\gamma_P$, $u \in P^{\alpha\gamma}$, or $u \in P^{\gamma\beta}$. If u equals α_P (resp. γ_P, β_P) then set $\hat{P}$ equal to α_P (resp. γ_P, β_P) and go to Step 4. Otherwise, if $u \in P^{\alpha\gamma}$ then do $\hat{P}:=P^{\alpha\gamma}$ and go to Step 4. Otherwise, if $u \in P^{\gamma\beta}$ then do $\hat{P}:=P^{\gamma\beta}$ and go to Step 4. Otherwise leave $\hat{P}$ unchanged. (An assignment like $\hat{P}:=P^{\alpha\gamma}$ is done in constant time simply by remembering the new first and last vertex of $\hat{P}$.)

Step 4. Use Proposition 4 to detect whether $w=\alpha_Q$, $w=\beta_Q$, $w=\gamma_Q$, $w \in Q^{\alpha\gamma}$, or $w \in Q^{\gamma\beta}$. If w equals α_Q (resp. γ_Q, β_Q) then set $\hat{Q}$ equal to α_Q (resp. γ_Q, β_Q) and go to Step 2. Otherwise, if $w \in Q^{\alpha\gamma}$ then do $\hat{Q}:=Q^{\alpha\gamma}$ and go to Step 2. Otherwise, if $w \in Q^{\gamma\beta}$ then do $\hat{Q}:=Q^{\gamma\beta}$ and go to Step 2. Otherwise leave $\hat{Q}$ unchanged.

(End of algorithm).

Correctness of the algorithm is easy to establish and its proof is therefore omitted.

Since every usage of Proposition 4 takes $O(c)$ time, the time complexity of the algorithm is equal to c multiplied by the number of times that steps 2-4 get executed. We now bound the number of times steps 2-4 are executed.

Lemma 2. Let a, b, $P^{\alpha\beta}$, $Q^{\alpha\beta}$, u, and w be as defined in algorithm **D**. Assume that $u \notin \{\alpha_P,\beta_P\}$ and $w \notin \{\alpha_Q,\beta_Q\}$. Then at least one of the following statements (a) or (b) is true

(a) $u \in P^{\alpha\beta}$,

(b) $w \in Q^{\alpha\beta}$.

Proof. Let T_a (resp. T_b) be the line perpendicular at a (resp. b) to the segment ab (see Figure 5). By the definition of a and b, T_a (resp. T_b) is tangent to P' (resp. Q'). Without loss of generality, T_a and T_b are vertical, P' is to the left of T_a, and Q' is to the right of T_b. If $u=a$ or $w=b$ then the lemma holds, so suppose that $u \neq a$ and $w \neq b$. By the definition of u and w, we must have $d(u,w) \leq d(a,b)$. This implies that u is to the right of T_a or w is to the left of T_b (possibly both). Hence, it suffices to show that if u (resp. w) is to the right (resp. left) of T_a (resp. T_b), then (a) (resp. (b)) holds. So suppose that u is to the right of T_a. We show that $u \in P^{\alpha\beta}$ (and hence (a) holds) by noting that, otherwise, when going from a to u (or u to a) along the boundary of $\hat{P}$, one encounters at least one vertex of P', let this vertex be a_r. Since $a_r \in P'$, it must be to the left of T_a (see Figure 6). Thus, in going from a to u along the boundary of $\hat{P}$ one starts with point a on the line T_a, then encounters point a_r to the left of T_a, then point u to the right of T_a. This contradicts the convexity of P. A similar argument shows that if w is to the left of T_b, then (b) holds. □

Corollary 3. Steps 2-4 of algorithm **D** are executed a total of at most $4c-1$ times.

Proofs. Lemma 2 implies that, every time we execute Steps 2-4, at least one of $\hat{P}$ or $\hat{Q}$ decreases in size by a factor of at least $s=n^{1/2c}$, thus proving the corollary. □

We have thus established the following

Theorem 3. Algorithm **D** correctly finds points $u \in P$ and $w \in Q$ such that $d(u,w)=d(P,Q)$. It uses $n^{1/c}$ processors and it runs in time $O(c^2)$, where c is any integer of our choice.

We now define a sequence of algorithms $\mathbf{E}_0, \mathbf{E}_1, \cdots$ in a manner entirely analogous to the way we defined sequence $\mathbf{B}_0, \mathbf{B}_1, \cdots$ in the previous section. The analysis, which is very similar to that done in Section 2 (and hence is omitted), establishes the following

Theorem 4. Let P and Q be two disjoint convex n-gons. Let $k=c^d$ where c and d are any integers, $c=O(1)$. A CREW-PRAM having $n^{1/k}$ processors can compute points $u \in P$ and $w \in Q$ such that $d(u,w)=d(P,Q)$ in $O(k^{1+\varepsilon(c)})$ time, where $\lim_{c \to \infty} \varepsilon(c) = 0$.

Corollary 4. Let P and Q be two disjoint convex n-gons. Let $k=c^d$ where c and d are any integers, $c=O(1)$. A CREW-PRAM having $n^{1/k}$ processors can compute a straight line L which separates P from Q in $O(k^{1+\varepsilon(c)})$ time, where $\lim_{c\to\infty}\varepsilon(c) = 0$.

It is worth pointing out that, given a straight line L and a convex polygon P, a method similar to that given above (actually an easier version of it) can be used to determine if L and P intersect or not, and if so to give a point of this intersection. This is done in time $O(k)$ if we have $n^{1/k}$ processors, where k is any integer of our choice. The details are left to the reader.

4. Conclusion

We gave new parallel algorithms for computing some functions of convex polygons in the CREW-PRAM model. In particular, we showed that constant time suffices for computing these functions even if we have a sublinear number of processors (in fact $n^{1/c}$ processors suffice for any constant integer c). Even with a linear number of processors, it was not previously known how to achieve constant time performance for computing these functions.

References

[1] M. J. Atallah and M. T. Goodrich, "Parallel Solutions to Geometric Problems," *Proc. 1985 IEEE International Conference on Parallel Processing*, St. Charles, Illinois, August 1985, pp. 411–417. (To appear in *J. of Parallel and Distributed Computing.*)

[2] A. Aggarwal, B. Chazelle, L. Guibas, C. O'Dunlaing and C. Yap, "Parallel Computational Geometry," *Proc. 26th Annual IEEE Symposium on Foundations of Computer Science*, October 1985, pp. 468-477.

[3] A. Chow, "Parallel Algorithms for Geometric Problems," Ph.D. dissertation, Computer Science Dept., University of Illinois at Urbana-Champaign, 1980.

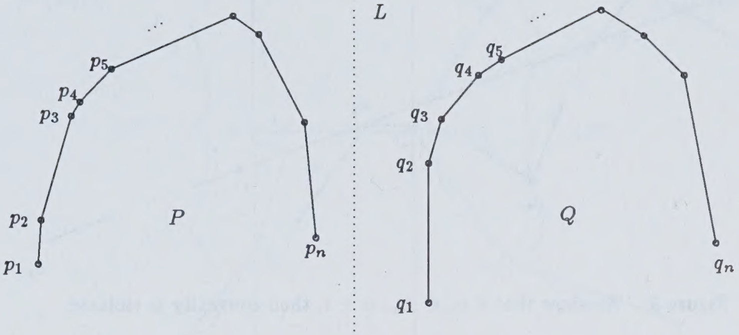
[4]. H. Edelsbrunner, "Computing the Extreme Distances between Two Convex Polygons," *J. Algorithms*, Vol. 6, 1985, 213-224.

[5] M.H. Overmars and J. Van Leeuwen, "Maintenance of Configurations in the Plane", *Journal of Computer and Systems Sciences*, Vol. 23, pp. 166-204 (1981).

[6] F. P. Preparata and S. J. Hong, "Convex Hulls of Finite Sets of Points in Two and Three Dimensions," *CACM*, Vol. 20, No. 2, February 1977, pp. 87-93.

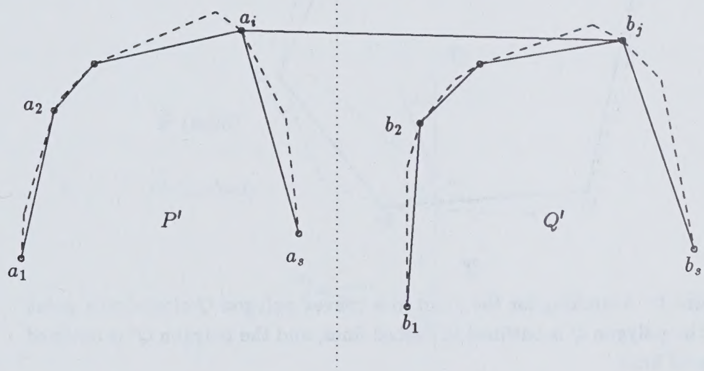
[7] F. P. Preparata and M. I. Shamos, *Computational Geometry, An Introduction,* Springer Verlag, New York, 1985.

Figures



L
p_1
p_2
p_3
p_4
p_5
p_n
P
q_1
q_2
q_3
q_4
q_5
q_n
Q

Figure 1: The two polygons P and Q. WLOG, the line separating P and Q is vertical and P is to the left of Q.



a_i
a_2
a_1
a_s
P'
b_j
b_2
b_1
b_s
Q'

Figure 2: The two subpolygons P' and Q' and their common supporting tangent a_ib_j. The polygons $\hat{P}$ and $\hat{Q}$ are shown in dashed lines.

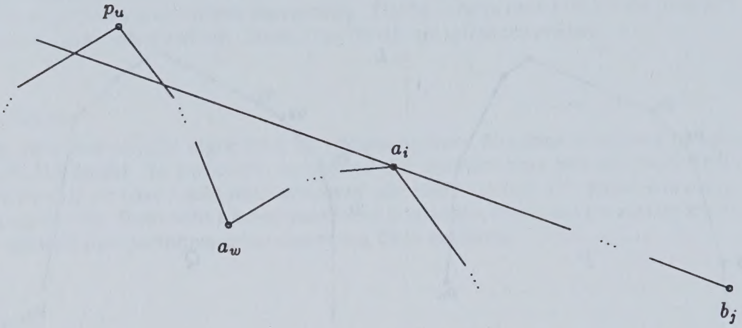


Figure 3: We show that if $p_u \in A_w$, $w < i$, then convexity is violated.

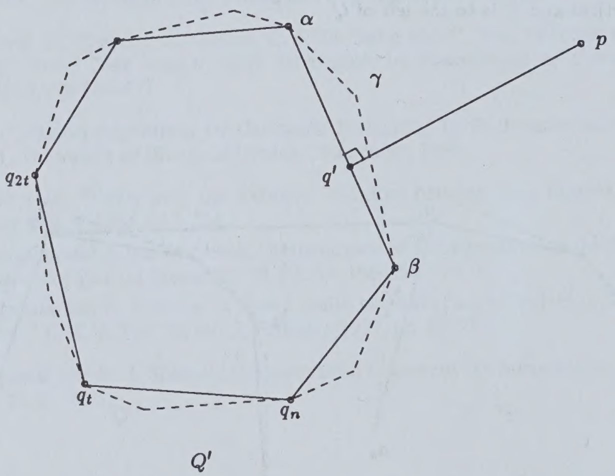

α
p
γ
q_{2t}
q'
β
q_t
q_n
Q'

Figure 4: Searching for the point in a convex polygon Q closest to a point p. The polygon Q is outlined in dashed lines, and the polygon Q' is outlined in solid lines.

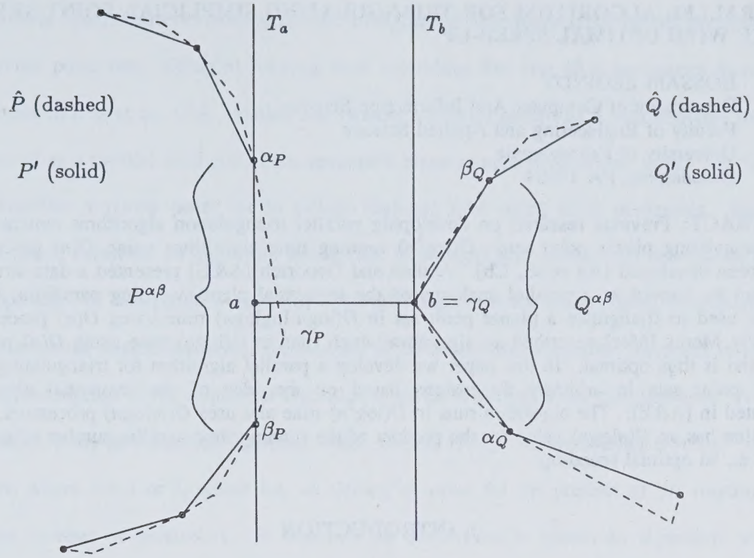

$\hat{P}$ (dashed)
P' (solid)
T_a
T_b
α_P
$P^{\alpha\beta}$
a
γ_P
β_P
$b = \gamma_Q$
β_Q
$\hat{Q}$ (dashed)
Q' (solid)
$Q^{\alpha\beta}$
α_Q

Figure 5: Reducing the size of $\hat{P}$ and/or $\hat{Q}$. $P^{\alpha\beta}$ is the (dashed) portion of $\hat{P}$ between α_P and β_P (inclusive), and γ_P subdivides $P^{\alpha\beta}$ into $P^{\alpha\gamma}$ and $P^{\gamma\beta}$. $Q^{\alpha\beta}$, $Q^{\alpha\gamma}$, and $Q^{\gamma\beta}$ are defined analougously.

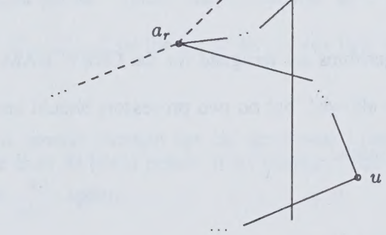

T_a
$\hat{P}$ (solid)
P' (dashed)
a
a_r
u

Figure 6: We show that if $u \notin P^{\alpha\beta}$, then convexity is violated.

A PARALLEL ALGORITHM FOR TRIANGULATING SIMPLICIAL POINT SETS IN SPACE WITH OPTIMAL SPEED-UP

HOSSAM ELGINDY
Department of Computer And Information Science
Faculty of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104

ABSTRACT: Previous research on developing parallel triangulation algorithms concnetrated on triangulating planar point sets. $O(\log^3 n)$ running time algorithms using $O(n)$ processors have been developed [Ag et al., Ch]. Atallah and Goodrich [A&G] presented a data structure that can be viewed as a parallel analogue of the sequential plane-sweeping paradigm, which can be used to triangulate a planar point set in $O(\log n \log\log n)$ time using $O(n)$ processors. Recently Merks [Mer] described an algorithm which runs in $O(\log n)$ time using $O(n)$ processors, and is thus optimal. In this paper we develop a parallel algorithm for triangulating simplicial point sets in arbitrary dimensions based on the idea of the sequential algorithm presented in [A&E]. The algorithm runs in $O(\log^2 n)$ time and uses $O(n/\log n)$ processors. The algorithm has an $O(n\log n)$ value for the product of the running time and the number of processors; i.e., an optimal speed-up.

1. INTRODUCTION

The problem of triangulating a finite set of points arises in numerical interpolation [L], in the finite element method [C], and as a preprocessing step for an efficient solution to the planar shortest path problem [As et al]. It has been shown that a triangulation of a set of n points in the plane can be constructed in $O(n\log n)$ time, which is optimal to within a constant factor. Optimality follows from a result of Shamos and Hoey [S&H] who have shown that any triangulation algorithm can also sort. Recenlty, Avis and ElGindy [A&E] presented a divide-and-conquer algorithm for triangulating simplicail point sets in two and higher dimensions. For a fixed dimension, the algorithm triangulates a simplicial set of n points in general position in $\Theta(n\log n)$ time.

Throughout this paper, algorithms are designed for the CREW RAM shared memory model in which concurrent reads are allowed, but no two processors should attempt to simultaneously write in the same memory location.

Previous research on developing parallel triangulation algorithms concnetrated on triangulating planar point sets. $O(\log^3 n)$ running time algorithms that use $O(n)$ processors have been developed in [Ag et al., Ch]. Atallah and Goodrich [A&G] presented a data structure that can be viewed as a parallel analogue of the sequential plane-sweeping paradigm, which can be used to triangulate a planar point set in $O(\log n \log\log n)$ time using $O(n)$ processors. Recently Merks [Mer] described an algorithm which runs in $O(\log n)$ time using $O(n)$ processors, and is thus optimal. In this paper we develop a parallel algorithm for triangulating simplicial point sets in arbitrary dimensions based on the idea of the sequential algorithm presented in [A&E]. A brief description of the sequential algorithm is given in section 2 for completeness. A direct immitation leads to a parallel algorithm which runs in $O(\log^{k+1} n)$ time using $O(n/\log^{k-1} n)$ processors, where $k = 1$ or 2, which has an $O(n\log^2 n)$ value for the product of the running time and the number of processors. In section 3 we show how to obtain an algorithm, with an optimal speed-up, that runs in $O(\log^2 n)$ time and uses $O(n/\log n)$ processors. The algorithm has an $O(n\log n)$ value for the product of the running time and the number of processors.

2. THE SEQUENTIAL TRIANGULATION ALGORITHM

Let **P** denote a set of points in $\mathbf{R}^d$ which has dimension d. **P** is in **general position** if each subset of **P** containing at most d+1 points is affinely independent. A simplex is a set of d+1 points in general position. A **triangulation** of **P** is a partition of the interior of the convex hull of **P** into simplices, the vertices of which are points of **P**. We say that **P** is **simplicial** if **P** has exactly d+1 extreme points. Unless stated otherwise, let **P** be a simplicial d-dimensional point set in $\mathbf{R}^d$ and $\{E_1, E_2, \ldots, E_{d+1}\}$ be the set of its convex hull vertices. Using the following geometric property:

> A simplicail set **P** in general position can be decomposed into d+1 simplicies, none of which contains more than $dn/(d+1)$ points in its interior. The decomposition vertex is a point of **P**, called the $\frac{dn}{d+1}$ splitter.

Avis and ElGindy [A&E] developed the following algorithm for triangulating **P**:

Algorithm TRIANGULATE (P)

input : A set P of n points.
output: The list of edges which form the traiangulation of P.
method:

Step 1. Find the $\frac{dn}{d+1}$ splitter of the set P as follows: define H as the hyperplane passing through $\{E_2, E_3, ..., E_{d+1}\}$. and find the $\frac{dn}{d+1}$*th* furthest point from H. Let y be such a point and let $P^{(1)}$ consist of the elements of P whose distance from H is less than or equal to that of the point y. We then project each point in $P^{(1)}$ onto H to compute a $d-1$ dimensional set, denoted by $P^{(2)}$ and find a $(d-1)|P^{(2)}|/d$ splitter of $P^{(2)}$, say z'. The point z of P, whose projection onto H is z', is the $\frac{dn}{d+1}$ splitter of P.

Step 2. Form $d+1$ simplices by joining z to all points in the set $\{E_1, E_2, ..., E_{d+1}\}$ and partition the set of points into sets $P_1, P_2, ..., P_{d+1}$.

Step 3. Recursively solve the triangulation problem for each of P_i, $i \in \{1,2,...,d+1\}$. After this recursive call returns we will have a complete triangulation of P.

End of Algorithm TRIANGULATE

Finding a splitter for P involves computing the hyperplane H and its normal vector h pointing towards E_1, reporting the $\lceil\frac{dn}{d+1}\rceil$*th* order statistic in the direction h, and projecting the points onto H and solving a $d-1$ dimensional problem with at most n points. The hyperplane H can be calculated $O(d^3)$ time, the one dimensional order statistic requires $O(n)$ time and the projections can be performed in $O(nd^3)$ time. Since we reduce the dimension by one in this time bound and we start in d dimensions, the overall running time for finding a splitter is $O(nd^4)$. Distributing the points between the $d+1$ simplices involves computing the relative position of each point with respect to $O(d)$ hyperplanes and can thus be performed in $O(nd^2)$ time.

Let n_i be the number of points lying in the interior of simplex S_i, and let $T(n,d)$ be the time required to triangulate a simplicial set of d dimensional points with n points in the interior of the convex hull. Therefore,

$$T(n,d) = \sum_{i=1}^{d+1} T(n_i,\ d) + cnd^4$$

where

$$\sum_{i=1}^{d+1} n_i = n-1 \ ,\ 0 \le n_i \le \frac{dn}{d+1} \ ,$$

and $T(0,d) = 1$. This recurrence has the solution

$$T(n,d) = O(nd^4 \log_{1+1/d} n\ ) \ .$$

A detailed running time analysis and proof of correctness can be found in [A&E].

3. THE PARALLEL TRIANGULATION ALGORITHM

Designing a parallel algorithm for triangulating a simplicial set of n points involves the following issues: (i) an efficient method for reporting the $\frac{dn}{d+1}$ splitter, and (ii) an efficient method for distributing the points into the $d+1$ simplices and the processors into the $d+1$ subproblems. We address each issue separately.

(i) Reporting A Splitter

The heart of the method for reporting the $\frac{dn}{d+1}$ splitter, described in section 2, involves reporting the $\frac{(d-i)n}{d+1-i}$*th* order statistic, $i = 0,1,\ldots,d-1$. A straight forward solution is to *sort* the distances from the hyperplane H and then report the point with the required rank. Therefore we can report the $\frac{dn}{d+1}$ splitter in $O(d^4 \log n)$ time using $O(n)$ processors [Le], or in $O(d^4 \log^2 n)$ time using $O(n/\log n)$ processors [S&V]. Since the number of points in each of the $d+1$ simplices is less than or equal to $\frac{dn}{d+1}$, it can be shown that the overall time is $O(g(d) \log^{k+1} n)$ using $O(n/\log^{k-1} n)$ processors, where $g(d) = d^4/\log(\frac{d+1}{d})$ and $k = 1$ or 2. For a fixed dimension, such an algorithm has an $O(n\log^2 n)$ value for the product of the running time and the number of processors, which is *not* an optimal speed-up.

By relaxing the requirement that the decomposition vertex must be the $\frac{dn}{d+1}$ splitter, we can develop a better method for reporting a "good" decomposition vertex which runs in $O(d^4 \log n)$ time using $O(n/\log n)$ processors. The reported vertex partitions **P** into $d+1$ simplices none of which contains more than αn points in its interior, where $\alpha \leq \frac{2^{2d}-1}{2^{2d}}$. The method to report such a vertex is as follows:

(i) Define H as the hyperplane passing through $E_2, E_3, \ldots, E_{d+1}$. For each point compute the vlaue of its distance from H.

(ii) Partition the points into $O(n/\log n)$ groups of $O(\log n)$ points each. For each group find the point with the middle vlaue sequentially using one processor, and then report the point with the middle value between the $O(n/\log n)$ middle values. Let y be such a point and let $\mathbf{P}^{(1)}$ consist of the elements of **P** whose distance from H is less than or equal to that of the point y.

(iii) If $d = 1$, then report y as the decomposition vertex of **P**. Otherwise we project each point in $\mathbf{P}^{(1)}$ onto H to compute a $d-1$ dimensional set, denoted by $\mathbf{P}^{(2)}$, and iteratively find a decomposition vertex of $\mathbf{P}^{(2)}$, say z'. Report the point z of **P**, whose projection onto H is z', as the decomposition vertex of **P**.

Lemma 1 The above method reports a vertex that partitions a simplicial set **P** with n points into $d+1$ simplices none of which contains more than αn points in its interior, where $\alpha \leq \frac{2^{2d}-1}{2^{2d}}$, in $O(d^4 \log n)$ time using $O(n/\log n)$ processors.

Proof. For $d = 1$, it is easy to show that the reported point paritions the set into two subsets none of which contains more than $3n/4$ points.

We now consider values of $d \geq 2$. The set $\mathbf{P}^{(2)}$ has dimension $d-1$. It is also simplicial with d extreme points $E_2, E_3, \ldots, E_{d+1}$ and contains $n' \geq n/4$ points. By induction, the above method reports z' as the $\frac{2^{2(d-1)}-1}{2^{2(d-1)}} n'$ decomposition vertex of the set $\mathbf{P}^{(2)}$. Consider the regions $S_i = \{E_1, \ldots, E_{i-1}, z, E_{i+1}, \ldots, E_{d+1}\}$, $i = 1, 2, \ldots, d+1$:

Case 1 ($i \geq 2$): S_i may contain points of $\mathbf{P}-\mathbf{P}^{(1)}$ together with points of $\mathbf{P}^{(1)}$ that were mapped into the region of H with extreme points $\{E_2, ..., E_{i-1}, z', E_{i+1}, ..., E_{d+1}\}$. Therefore

$$
\begin{aligned}
|S_i| &\leq n - n' + \frac{2^{2(d-1)}-1}{2^{2(d-1)}} n' \\
&= n - \frac{n'}{2^{2(d-1)}} \\
&\leq n - \frac{1}{2^{2(d-1)}} \frac{n}{4} \\
&= \frac{2^{2d}-1}{2^{2d}} n
\end{aligned}
$$

Case 2 ($i = 1$): By construction, the points interior to S_1 are contained in $\mathbf{P}^{(1)}$. By the choice of y this set has at most $3n/4$ elements, which is less than or equal to $\frac{2^{2d}-1}{2^{2d}} n$ for all d.

It remains to analyse the running time of the above method. Reporting the point y, in step (ii), involves sequentially finding the median of $O(\log n)$ numbers, and finding the median of $O(n/\log n)$ numbers using $O(n/\log n)$ processors. It is easy to see that both operations can be performed in $O(\log n)$ time [A&H&U, Le]. The projection of $\mathbf{P}^{(1)}$ onto the hyperplane H, in step (iii), can be performed in $O(d^3 \log n)$ time. Since we reduce the dimension by one in this time bound and we start in d dimensions, the overall running time for finding a decomposition vertex is $O(d^4 \log n)$ time using $O(n/\log n)$ processors. ■

We conclude the discussion of this issue by pointing out that the running time of the presented method remains unchanged, i.e. $O(\log n)$, when it is used to report the splitter of γn points using $O(\gamma n/\log n)$ processors, where γ is an arbitrary constant that is less than one.

(ii) Distributing the points and the processors

The process of distributing the set **P** between the $d+1$ simplices involves computing the relative position of each point with respect $O(d)$ hyperplanes. Using $O(n/\log n)$ processors, it can be shown that $O(d^2 \log n)$ is sufficient to distribute the points between the $d+1$ simplices and to count the number of points in each of the $d+1$ sub-regions. Let $\gamma_i n$ be the number of points to be found in the *i th* sub-region, $i = 1, 2,..., d+1$. We then allocate $\gamma_i n/\log n$ of the processors to recursively manipulate the points in the *i th* sub-region.

We can now state the parallel algorithm for triangulating simplicial points

Algorithm P-TRIANGULATE (P, | P | / log*n*)

```
    if $|P| \le \log n$ then
        triangulate P sequentially using TRIANGULATE (P)
    else
        find a decomposition vertex of the set P.

        partition the points into $P_1, . . . , P_{d+1}$ which correspond to the $d+1$ sub-regions, and
        count the number of elements in each subset.

        P-TRIANGULATE ($P_i$, $| P_i |$ / $\log n$),    $i = 1, 2,..., d+1$
End of Algorithm P-TRIANGULATE
```

Theorem 1 Algorithm **P-TRIANGULATE** triangulates a *d*-dimensional simplicial point set in

$$O(f(d)\log^2 n) \text{ using } O(n/\log n) \text{ processors, where } f(d) = d^4/\log(\frac{2^{2d}}{2^{2d}-1}).$$

Proof. The decomposition vertex paritions **P** into $d+1$ simplices none of which contains more than $\frac{2^{2d}-1}{2^{2d}} n$ points. Therefore, the running time and the required number of processors can be expressed as follows:

$$T(n,d) = max \;\; \{ \; T(n', d) + cd^4 \log n \;\; , \;\; \log n \, \log\log n \; \}$$

$$Proc(n/\log n, d) = max \ \{ \ \sum_{i=1}^{d+1} Proc(n_i/\log n,\ d) \ , \ n/\log n \ \}$$

where

$$\sum_{i=1}^{d+1} n_i = n-1 \ , \ 0 \le n_i \le \frac{2^{2d}-1}{2^{2d}} n \ , \ n' = \max_{i=1,..,d+1} \ \{ n_i \}$$

The above recurrences have the following solutions:

$$T(n,d) = O(f(d)\log^2 n) \ , \ where \ f(d) = d^4/\log(\frac{2^{2d}}{2^{2d}-1})$$

$$Proc(n/\log n,d) = O(n/\log n)$$

which completes the proof. ■

We conclude this section by pointing out that, for a fixed dimension, the algorithm has an optimal value for the product of the running time and the number of processors which is $\Theta(n\log n)$.

4. CONCLUDING REMARKS

We have presented a parallel algorithm for triangulating simplicial point sets which runs in $O(\log^2 n)$ time using $O(n/\log n)$ processors for any fixed dimension. The algorithm achieves optimal speed-up by relaxing the requirement that the decomposition vertex must be the ''best'' splitter (i.e., $\frac{dn}{d+1}$ splitter) to just being a ''good'' splitter (i.e., $\frac{(2^{2d}-1)n}{2^{2d}}$ splitter in the worst-case). Therefore we can develop an algorithm for reporting a decomposition vertex with an optimal speed-up. Merks [Mer] has shown that the optimal running time of $O(\log n)$ can be achieved for planar sets using $O(n)$ processors. It is still an open problem whether such running time can be achieved for three and higher dimensions.

REFERENCES

[Ag et al.] A. Aggarwal, B. Chazelle, L. Guibas, C. O'Dunlaing, and C. Yap, "Parallel Computational Geometry: extended abstract," *Proceedings of 26th Annual Symposium on Foundations of Computer Science*, Portland, October 1985, pp. 468-477.

[A&H&U] A.V. Aho, J.E. Hopcroft and J.D. Ullman, *The Design and Analysis of Computer Algorithms*, Readings, Addison-Wesley Publishing Company, 1974.

[As et al.] T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, "Visibility polygon search and euclidean shortest paths," *Proceedings of 26th Annual Symposium on Foundations of Computer Science*, Portland, October 1985, pp. 155-164.

[A&G] M.J. Atallah and M.T. Goodrich, "Efficient plane sweeping in parallel," *Proceedings of the Second Symposium on Computational Geometry*, York Heights, June 1986.

[A&E] D. Avis and H. ElGindy, "Triangulating simplicial point sets in space: extended abstarct," *Proceedings of the Second Symposium on Computational Geometry*, York Heights, June 1986.

[C] J.C. Cavendish, "Automatic triangulation of arbitrary planar domains for the finite element method," *Int. J. Numer. Methods Eng.*, vol.8 (1974), pp. 679-696.

[Ch] A. Chow, *Parallel Algorithms for Geometric Problems.* Ph.D. dissertation, Computer Science Department, University of Illinois at Urbana-Champaign, 1980.

[L] C.L. Lawson, "C^1-compatible interpolation over a triangle," *Jet Propulsion Lab.*, Technical Memo. 33-770, May 1976.

[Le] T. Leighton, "Tight bounds on the complexity of parallel sorting," *IEEE Transactions on Computers*, vol.C-34 (1985), no.4, pp. 344-354.

[Mer] E. Merks, "Triangulating a set of points in the plane," manuscript, Department of Computing Science, Simon Fraser University, April 1986.

[S&H] M.I. Shamos and D. Hoey, "Closest point problems," *Proceedings of 16th Annual Symposium on Foundations of Computer Science*, October 1975, pp. 151-162.

[S&V] Y. Shiloach and U. Vishkin, "Finding the maximum, merging, and sorting in a parallel computation model," *Journal of Algorithms*, vol.2 (1981), pp. 88-102.

EFFICIENT PARALLEL ALGORITHMS FOR SOLVING SOME TREE PROBLEMS

XIN HE

Department of Computer and Information Science
The Ohio State University, Columbus, Ohio.

ABSTRACT

We define the binary tree algebraic computation (BTAC) problem. By generalizing the "tree contraction" scheme of Miller and Reif [5], a parallel algorithm for the BTAC problem is obtained. We show that a variety of graph problems (maximum independent set, minimum r-dominating set, maximum matching set etc.) on trees can be converted to instances of the BTAC problem. Thus the efficient parallel algorithms for these problems are obtained systematically by using the BTAC algorithm. The parallel computation model is an exclusive read exclusive write PRAM. Our algorithms for solving tree problems run in $O(logn)$ ($O(rlogn)$ for the r-dominating set problem) time with $O(n)$ processors. As another application of the BTAC algorithm, we show that any algebraic expression on any finite set can be evaluated in $O(logn)$ time with $O(n)$ processors.

1. INTRODUCTION

We define the *binary tree algebraic computation* (BTAC) problem and generalize the "parallel tree contraction" method of Miller and Reif [5] to obtain an efficient parallel algorithm for solving this problem. A variety of graph problems on trees can be converted to instances of the BTAC problem and solved by using the BTAC algorithm. These problems include: minimum covering set (MCS), maximum independent set (MIS), maximum matching set (MMS) and minimum r-dominating set (r-MDS) problems. The linear time sequential algorithms for these problems exist ([3,4,6]), but efficient parallel algorithms for them have not been studied. Our parallel algorithms run in $O(logn)$ time ($O(rlogn)$) for the r-MDS problem) with $O(n)$ processors.

The parallel computation model used in this paper is an *exclusive read exclusive write parallel random access machine* (PRAM). The model consists of a number of identical processors and a common memory. In each time unit, a processor can read from a memory cell, perform an ordinary arithmetic or a logic computation and write into a common memory cell. Read from or write into the same memory cell by different processors at the same time are not allowed. This model has been widely used in the literature for implementing parallel algorithms.

In section 2, the BTAC problem is defined and a parallel algorithm for solving this problem is presented. In section 3, the BTAC algorithm is applied to solve the MCS, MIS and MMS problems. In section 4, we discusses the r-MDS algorithm.

2. BINARY TREE ALGEBRAIC COMPUTATION PROBLEM

Let $S=\{s_1, \cdots, s_k\}$ be a finite set. Let $\Psi_S=\{g:S\times S\to S\}$ be the set of all binary functions on S. A ***binary computation tree*** on S is a binary tree T. Each leaf v of T is labeled by an element $L(v)\in S$. Each internal vertex u of T is labeled by a function $F(u)\in\Psi_S$. The ***algebraic computation*** on T is defined as follows. Suppose u is an internal vertex of T whose two children v_1,v_2 are leaves. Compute $L(u)=F(u)(L(v_1),L(v_2))$, assign $L(u)$ as the final label of u, and delete v_1,v_2. (Thus u becomes a new leaf). Repeat this operation until all vertices of T get a final label $L(u)\in S$. The ***BTAC problem on S*** is to perform this computation on arbitrary T. Such a tree T corresponds to an algebraic expression on S in the obvious way. The BTAC problem is equivalent to the evaluation of the expression and all sub-expressions in it. If $S=\{0,1\}$, we are dealing with the boolean expressions in which any of the 16 boolean functions may occur.

Theorem 1: Let S be any set of size k. Assuming the input is given by the parent pointers, the BTAC problem on S can be solved in $O(k\log n)$ time with $O(n)$ processors on a PRAM, where n is the number of vertices in the input computation tree.

Proof: Our algorithm is a generalization of the "tree contraction" scheme in [5]. One application of this method in [5] is to evaluate arithmetic computation trees where internal vertices are labeled by + or ×. In our case, the internal vertices are labeled by arbitrary functions on S which may be neither commutative nor associative.

The "tree contraction" method makes uses of two operations: RAKE and COMPRESS. The RAKE operation deletes all leaves of T. Suppose u is an internal vertex of T and v_1,v_2 are its two children. If both v_1,v_2 are leaves, they are deleted. The final label $L(u)=F(u)(L(v_1),L(v_2))$ can be computed in constant time by one processor. (By looking up the table for the function $F(u)$). Suppose only one child of u, say v_2, is a leaf. Knowing one of its argument $L(v_2)$, u represents an unary function: $F^1(u)=F(u)(*,L(v_2)):S\rightarrow S$. Since $|S|=k$, $F^1(u)$ can be represented as a k-element vector $(s_{i_1}, \cdots, s_{i_k})$ such that $F^1(u)(s_j)=s_{i_j}$. This vector can be computed by one processor in $O(k)$ time and is stored in some common memory locations associated with u.

If T is highly unbalanced, it may take $O(n)$ RAKE operations to compute $L(u)$ for all vertices. The COMPRESS operation is introduced to solve this problem. A *chain* of T is a sequence of vertices $C=v_1 \cdots v_l$ where v_i is the only child of v_{i+1}. C is a *maximal chain* if it is a chain and is not properly contained in any longer chains. The COMPRESS operation shortens all maximal chains $C=v_1 \cdots v_l$ $(l \geq 3)$ of T. More precisely, for each even i, it deletes v_i and makes v_{i+1} the parent of v_{i-1}. Before this operation, each vertex v_i in C represents an unary function $F^1(v_i)$. In order to avoid changing the subsequent evaluation, we associate v_{i+1} with the composite function $F^1(v_{i+1}) \circ F^1(v_i)$. This new function, still in the vector form, can be computed from the two vectors for $F^1(v_{i+1})$ and $F^1(v_i)$ in $O(k)$ time by one processor. In this operation, vertices v_i (i even) are deleted from T without their final labels having been computed. However, $L(v_i)$ depends only on $L(v_{i-1})$ and the function $F^1(v_i)$. We associate each vertex v with a $stack(v)$. Before deleting v_i, the vector for $F^1(v_i)$ is pushed on $stack(v_{i-1})$. After $L(v_{i-1})$ is available, $F^1(v_i)$ is popped from $stack(v_{i-1})$ and $L(v_i)=F^1(v_i)(L(v_{i-1}))$ can be computed in constant time by one processor.

The BTAC algorithm consists of two phases. The first phase repeatedly applies the RAKE and the COMPRESS operation on T until T is reduced to its root. [5] shows that this can be done by at most $\log_{5/4} n$ iterations. In the second phase, all $stack(v)$ where $L(v)$ have been computed are popped in parallel. The final labels which have not been computed in the first phase are computed. Since the maximal depth of these stacks is at most $O(logn)$, all vertices will get their final labels in $O(logn)$ steps.

To implement this parallel algorithm, each vertex v of T is assigned a processor P_v. The input is given by the parent pointers $P(v)$, and the initial labels $L(v)$ or $F(v)$ for each vertex v. These values are stored in common memory locations associated with v. In addition, each v is associated with some common memory locations to store $F^1(v)$, $stack(v)$ as discussed above. In the RAKE and the COMPRESS operation, each processor P_v modifies the memory locations associated with v or the parent of v. (See [5] for details). Since each operation can be executed in $O(k)$ time as discussed above, the algorithm takes $O(klogn)$ time and $O(n)$ processors. □

Corollary 2: Any algebraic expression on any set S of size k can be evaluated in $O(k \log n)$ time with $O(n)$ processors on a PRAM, where n is the total number of operands and operators in the expression. □

In order to solve graph problems for arbitrary trees, we remove the restriction that T must be a binary tree. A *computation tree* T on S is similar to a binary computation tree, except that internal vertices of T may have more than two children. Suppose u is an internal vertex of T with a function label $F(u)$ and $v_1, \cdots, v_l$ are children of u with final labels $L(v_i)$ $(1 \le i \le l)$, respectively. The final label $L(u)$ is obtained by applying $F(u)$ on $L(v_1) \cdots L(v_l)$ from the left to the right. (That is, apply $F(u)$ on $L(v_1), L(v_2)$, then apply $F(u)$ on the result and $L(v_3)$ $\cdots$ and so on). The *tree algebraic computation* (TAC) problem on S is to compute the final labels for all vertices of T.

Theorem 3: For any set S of size k, the TAC problem on S can be solved in $O(k \log n)$ time with $O(n)$ processors on a PRAM, assuming the input is given by the parent pointers.

Proof: Let T be a computation tree on S. By a parallel sorting, we can compute the adjacency list of T and determine the number of children for each internal vertex u. The sorting can be done in $O(\log n)$ time with $O(n)$ processors on a PRAM (See [1]). Suppose u has children $v_1, \cdots, v_l$ $(l \ge 3)$, listed from the left to the right. We add $l-2$ dummy vertices $u_2, \cdots, u_{l-1}$, and give each of them a label $F(u_i) = F(u)$. We make u_i the left child of u_{i+1}, $(2 \le i \le l-1$, where $u_l = u)$, make v_1 the left child of u_2, v_i the right child of u_i $(2 \le i \le l, u_l = u)$. After performing this operation on all internal vertices of T, we get a binary computation tree T_1. Clearly, $|T_1| \le 2|T|$. ($|T|$ denotes the number of vertices in T). After performing the BTAC algorithm on T_1, the final labels of the original vertices of T is exactly the same as the final labels they get after performing the tree algebraic computation on T. On a PRAM with $O(n)$ processors, the conversion from T to T_1 can be done in constant time (after sorting). The BTAC algorithm performed on T_1 takes $O(k \log n)$ time. This algorithm solves the TAC problem within the processor and time bounds stated above. □

3. ALGORITHMS FOR MCS, MIS AND MMS PROBLEMS

We apply the TAC algorithm to solve the MCS, MIS and MMS problems on trees. We assume the input is given by the parent pointers. In this section, V or $V(T)$ denotes the vertex set of T. E or $E(T)$ denotes the edge set of T.

A *covering set* of T is a subset $C \subseteq V$ such that for any edge $(i,j) \in E$, $\{i,j\} \cap C \neq \emptyset$. The MCS problem is to find a covering set of minimum cardinality. We actually solve the following slightly more general problem. Given a tree T, a *labeling* on T is a mapping $L: V \to \{F,R\}$. A *mixed covering* (mc for short) set of T with respect to L is a subset $C \subseteq V$ such that C is a covering set of T and for any $v \in V$, $L(v)=R$ implies $v \in C$. Let $mc(T,L)$ denote the cardinality of a minimum mc set of T with respect to L. By this definition, a covering set of T is just a mixed covering set of T with respect to the labeling L where $L(v)=F$ for all v.

Lemma 4: Let $T=(V,E)$ be a tree with a labeling L. Suppose v is a leaf of T and u is its parent. Define $T' = T - \{v\}$.

(a) if $L(v)=F$, let L' be the labeling of T' defined as: $L'(w)=L(w)$ for $w \in V-\{u,v\}$ and $L'(u)=R$, then $mc(T,L)=mc(T',L')$.

(b) if $L(v)=R$, let L' be the restriction of L on T', then $mc(T,L)=mc(T',L')+1$.

Proof: We only prove (b). The proof of (a) is similar. Let C' be a minimum mc set of T' with respect to L'. Define $C=C' \cup \{v\}$. For any $w \in V-\{v\}$, $L(w)=R$ implies $L'(w)=R$. This in turn implies $w \in C' \subseteq C$. Since v is also in C, $L(w)=R$ implies $w \in C$ for any $w \in V$. Let (i,j) be an edge of T. If $(i,j) \neq (v,u)$, (i,j) is also an edge of T'. So $\{i,j\} \cap C \neq \emptyset$. If $(i,j)=(v,u)$, we trivially have $\{i,j\} \cap C \neq \emptyset$. So C is a mc set of T with respect to L. Thus $mc(T,L) \leq |C| = |C'|+1 = mc(T',L')+1$.

Let C be a minimum mc set of T with respect to L. Let $C' = C-\{v\}$. One can similarly show that C' is a mc set of T' with respect to L'. So $mc(T',L') \leq |C'| = |C|-1 = mc(T,L)-1$. □

x	F	R
F	R	F
R	R	R

Fig 1

Let $S=\{F,R\}$. Define a binary function f on S as in Fig 1 in which the row (column) index is the first (second) argument. Let T be a tree with a labeling L. Suppose v is a leaf of T and u is its parent. If $L(u)=x$ and $L(v)=y$, $f(x,y)$ is the label received by u after deleting v from T as in lemma

4. Lemma 4 suggests the following sequential algorithm for computing a minimum mc set of T with respect to L: Let u be an internal vertex of T whose children $v_1 \cdots v_l$ are leaves. Apply f on $L(u), L(v_1), \cdots, L(v_l)$ from the left to the right, (i.e. apply f on $L(u)$ and $L(v_1)$, then apply f on the result and $L(v_2)$, $\cdots$ and so on). Assign the result to u as its final label and delete v_i $(1 \le i \le l)$. Repeat this operation until all vertices of T get a final label. A minimum mc set of T with respect to L will be $\{v \mid$ the final label of v is $R\}$.

This computation can be converted to an instance of the TAC problem as follows. For each internal vertex u of T, add a dummy vertex u_1, make u_1 the leftmost child of u. Label u_1 by the initial label of u, and re-label u by the function f. This gives a computation tree T_1 on S. This conversion can be done in constant time with $O(n)$ processors on a PRAM. After performing the TAC algorithm on T_1, the final labels on original vertices of T are exactly the final labels needed to identify a minimum mc set of T. So a minimum mc set (in particular, a MCS) of T can be computed in $O(logn)$ time with $O(n)$ processors on a PRAM.

An *independent set* of T is a subset $I \subseteq V$ such that no two vertices in I are adjacent. A MIS is an independent set of the maximum cardinality. It is well known that $C \subseteq V$ is a MCS of T iff $V-C$ is a MIS of T [2]. So the MCS algorithm can be applied to solve the MIS problem.

A *matching set* of T is a subset $M \subseteq E$ such that no two edges in M share a common vertex. The MMS problem is to find a matching set with maximum number of edges. Our MCS algorithm can be used to solve the MMS problem.

Lemma 5: Let T be a tree with all vertices labeled by F. Let $L(v)$ $(v \in V)$ be the final labels computed by the MCS algorithm.

(a) Suppose u is an internal vertex of T and $v_1, \cdots, v_k$ are its children. If $L(u)=R$, at least one v_i has a final label $L(v_i)=F$;

(b) For each u with $L(u)=R$, select a child v_{i_u} of u with a final label F, then the collection of edges $M=\{\ (v_{i_u}, u) \mid L(u)=R\ \}$ is a MMS of T.

Proof : (a) is obviously true from the algorithm.

(b) M is clearly a matching. It is known that in a bipartite graph, the cardinality of a MCS equals to the cardinality of a MMS [2]. Thus M is a maximum matching. □

This suggests an algorithm for computing a MMS of T: label all vertices of T by F, apply the TAC algorithm to compute the final labels $L(v)$. For each vertex u with a final label R, select a child with a final label F. This selection can be done in $O(logn)$ time with $O(n)$ processors on a PRAM.

In summary:

Theorem 6: If the input is given by the parent pointers, the MCS, MIS and MMS problems on trees can be solved in $O(logn)$ time with $O(n)$ processors on a PRAM.

4. ALGORITHM FOR r-MDS PROBEM

For $u,v \in V(T)$, let $d_T(u,v)$ denote the distance between u and v in T. If $d_T(u,v) \leq r$, we say v is *r-dominated by* u. A *r-dominating set* of T is a subset $D \subseteq V$ such that any vertex v of T is r-dominated by some vertex $u \in D$. The r-MDS problem is to find a r-dominating set in T with minimum cardinality. The sequential algorithms for r-MDS problem have been developed in [3], [4] and [6]. We first describe a sequential algorithm for the problem which is similar to the algorithms in [4] and [6], but easier to be parallelized.

Define $S=\{F_1, \cdots, F_r, B_1, \cdots, B_r, R\}$. Let T be a tree with a labeling $L: V \to S$. A *mixed r-dominating set* (md set for short) of T with respect to L is a subset $D \subseteq V$ such that:

(1) $L(v)=R$ implies $v \in D$;

(2) $L(v)=B_i$ implies that either there exists an $u \in V$ such that $L(u)=F_j$ and $j+d_T(u,v) \leq i$; or there exists an $u \in D$ such that $d_T(u,v) \leq i$. In either case, we say v *is bound to* u in D.

(3) A vertex v with $L(v)=F_i$ need not be included in D, but may be included in D in order to bound the vertices with label B_j.

By this definition, a r-dominating set of T is just a md set of T with respect to the labeling L where $L(v)=B_r$ for all $v \in V$. Let $md(T,L)$ denote the cardinality of a minimum md set of T with respect to L.

Define a binary function $f: S \times S \to S$ as follows: (in these formulas $B_0=R$).

(4.1) $f(R,R)=R$,

(4.2) $f(X,R)=F_1$, for any $X \in S$, $X \neq R$;

(5) For each B_j:

(5.1) $f(R,B_j)=R$,

(5.2) $f(B_i,B_j)=B_{j-1}$, for $i \geq j$,

(5.3) $f(B_i,B_j)=B_i$, for $i<j$,

(5.4) $f(F_i, B_j) = F_i$, for $i < j$,

(5.5) $f(F_i, B_j) = B_{j-1}$, for $i \geq j$;

(6) For each F_j:

(6.1) $f(R, F_j) = R$,

(6.2) $f(B_i, F_j) = F_{j+1}$, for $i > j$,

(6.3) $f(B_i, F_j) = B_i$, for $i \leq j$,

(6.4) $f(F_i, F_j) = F_{j+1}$, for $i > j$,

(6.5) $f(F_i, F_j) = F_i$, for $i \leq j$.

×	F1	F2	B1	B2	R
F1	F1	F1	R	F1	F1
F2	F2	F2	R	B1	F1
B1	B1	B1	R	B1	F1
B2	F2	B2	R	B1	F1
R	R	R	R	R	R

Fig 2

The function f for $r = 2$ is shown in Fig 2. The row (column) index is the first (second) argument. We next describe a linear time sequential algorithm for computing a minimum r-dominating set of T. First label all vertices v of T by $L(v) = B_r$. Suppose v is a leaf of T with a label $L(v)$ and u is its parent with a label $L(u)$. Delete v and re-label u by $f(L(u), L(v))$. Repeat this operation until all vertices of T get a final label. Upon the termination, a minimum r-dominating set of T is either:

(a) $D = \{u \mid \text{the final label of } u \text{ is } R\}$ if the final label of the root t of T is F_j for some j; or

(b) $D = \{u \mid \text{the final label of } u \text{ is } R\} \bigcup \{t\}$ otherwise.

Intuitively, if a vertex u gets a final label R, u must be included in D. If u gets a final label F_i, u is i-dominated by a vertex $v \in D$ from "below". (i.e. v has a final label R, v is a descendent of u and $d_T(u,v) \leq i$). If u has a final label B_i, u must be i-dominated by a vertex $v \in D$ from "above". (i.e. v has a final label R, v is not a descendent of u and $d_T(u,v) \leq i$). The correctness of this sequential algorithm follows from:

Lemma 7: Let T be a tree with a labeling $L: V \to S$. Suppose v is a leaf of T and u is its parent. Let $T' = T - \{v\}$. Define a labeling L' on T' by: $L'(u) = f(L(u), L(v))$ and $L'(w) = L(w)$ for other vertex w in T'. Then

(a) if $L(v) = R$, $md(T,L) = md(T',L') + 1$;

(b) if $L(v) \neq R$, $md(T,L) = md(T',L')$.

Proof: (a) Let D be a minimum md set of T with respect to L. Define $D' = D - \{v\}$.

If $L(u) = R$, $L'(u) = R$, D' is clearly a md set of T' with respect to L'.

If $L(u) \neq R$, $L'(u) = F_1$, consider any vertex $w \neq u$ in T' with $L'(w) = B_i$. If w is bound to v in D, we have $d_T(w,v) \leq i$. So $d_{T'}(w,u) + 1 = d_T(w,v) \leq i$, w is bound to u in D'. Thus, D' is a md set of T' with respect to L'. So $md(T',L') \leq |D'| = |D| - 1 = md(T,L) - 1$.

Now let D' is a minimum md set of T' with respect to L'. Define $D = D' \bigcup \{v\}$.

If $L(u) = R$, $L'(u) = R$, D is clearly a md set of T with respect to L.

If $L(u) \neq R$, $L'(u) = F_1$, consider any vertex w of T with $L(w) = B_i$. If w is bound to u in D', we have $d_{T'}(w,u) + 1 \leq i$. So $d_T(w,v) = d_{T'}(w,u) + 1 \leq i$, w is bound to v in D. Thus, D is a md set of T with respect to L. So $md(T,L) \leq |D| = |D'| + 1 = md(T',L') + 1$.

(b) Let D be a minimum md set of T with respect to L. Let $D' = D$. There are 10 cases (by formulas (5.1)-(5.5) and (6.1)-(6.5)), we only consider one of them (the case defined by (6.3)), other cases are similar. Suppose $L(u) = B_i$, $L(v) = F_j$, $(i \leq j)$. By (6.3), $L'(u) = f(B_i, F_j) = B_i$. In D, suppose u is bound to a vertex x with $L(x) = F_l$ (if $L(x) = R$, the proof is similar). Namely, $l + d_T(x,u) \leq i$. By our assumption, x can not be v. So x is a vertex in T'. Consider any vertex w in T' with $L'(w) = L(w) = B_k$. If w is bound to v in D, i.e. $j + d_T(v,w) \leq k$, we have: $l + d_T(x,w) \leq l + d_T(x,u) + d_T(u,w) \leq i + d_T(u,w) \leq j + d_T(u,w) < j + d_T(v,w) \leq k$. So w is bound to x in D'.

Thus, D' is a md set of T' with respect to L'. So $md(T',L') \leq |D'| = |D| = md(T,L)$.

Now let D' be a minimum md set of T' with respect L'. We can similarly prove that $D = D'$ is a md set of T with respect to L. Thus, $md(T,L) \leq |D| = |D'| = md(T',L')$. □

As in section 3, the sequential algorithm described above can be converted to an instance of the TAC problem and solved by the TAC algorithm. Since S has size $2r+1$, we conclude:

Theorem 8: The r-MDS problem for trees can be solved by a PRAM in $O(rlogn)$ time with $O(n)$ processors. □

The time bound of the r-MDS algorithm is proportional to r. If r is a small number (say a constant independent of n), our algorithm is quite efficient. However, if r is a large number (say $O(logn)$), the algorithm becomes less efficient. In a subsequent paper, the more general cases of the minimum r-dominating set problem are considered. The time bound is reduced to $O(loglognlogn)$ with $O(n)$ processors on a PRAM.

References:

[1]: M. Ajtai, K. Komlos, and E. Szmeredi, An $O(nlogn)$ sorting network, Proc. 15th ACM STOC, 1983, pp.1-9.

[2]: J. A. Bondy, U. S. R. Murty, Graph Theory with Applications, North Holland, New York, 1980.

[3]: E. Cockayne, S. Goodman and S. Henetniemi, A Linear Algorithm for the Domination Number of a Tree, Information processing letters, vol. 4, 1975, pp.41-44.

[4]: O. Kariv and S. L. Hakimi, An Algorithmic Approach on Network Location Problems I: p-Centers, SIAM J. Appl. Math., vol. 37, no. 3, 1979, pp. 513-538.

[5]: Gary L. Miller, John H. Reif, Parallel Tree Contraction and its Applications, Proc. 26th FOCS, 1985, pp. 478-489.

[6]: P. J. Slater, R-Domination in Graphs, J. ACM, vol. 23, no. 3, 1976, pp. 446-450.

PARALLEL MATROID INTERSECTION

ADA FU and JOSEPH G. PETERS
School of Computing Science
Simon Fraser University
Burnaby, British Columbia, Canada V5A 1S6

ABSTRACT

Parallel algorithms which find a maximum independent set in the intersection of two arbitrary matroids are presented. For both the weighted and unweighted versions of the problem, algorithms with depth $O(R(\log R + c(m)))$ are given, where m is the number of elements in the matroids, and R is the minimum of the ranks of the two matroids. c(m) is the time to test for independence and depends on the structures of the particular matroids. The unweighted algorithm uses $m^3/\log R$ processors and the weighted algorithm uses m^3 processors. The model of computation is a CRCW PRAM with two additional primitives similar to *fetch-and-add*. The serial time bounds for the unweighted and weighted problems respectively are $O(mR^2c(m))$ and $O(mR^2c(m)+mR^3)$. Both problems are log-space complete for P, so further dramatic decreases in depth are very unlikely.

1. INTRODUCTION

The class of combinatorial optimization problems based on matroids is large and includes many important problems as special cases. Problems involving a single matroid, such as the minimum weight spanning tree problem, are solved optimally by greedy algorithms. Many fast parallel algorithms for finding minimum weight spanning trees (or forests) have been published. See [8] for a brief survey. Cook [2] describes a fast parallel greedy algorithm for arbitrary matroids which generalizes an algorithm in [1]. The problems of finding a maximum cardinality or maximum weight independent set in the intersection of two matroids are also solvable in polynomial time. However, both problems are log-space complete for P [5], so it is very unlikely that poly-log depth parallel algorithms with polynomial width exist for these problems. The weighted and unweighted intersection problems for three or more matroids are easily shown to be NP-hard ([9], [6]).

Problems involving the intersection of two matroids include bipartite matching, finding maximum weight directed spanning trees (i.e., optimal branchings), and a variety of network flow problems. Few parallel algorithms have been found for these problems. Dekel and Sahni [4] give an $O(\log^2 n)$ algorithm (for an SIMD model) for the special case of matching in *convex* bipartite graphs and Shiloach and Vishkin [11] describe an $O(n^3(\log n)/p)$ algorithm, where $p \leq n$ is the number of processors, for finding a maximum flow in a directed flow network. This paper contains parallel algorithms for the weighted and unweighted intersections of two arbitrary matroids. We believe that these are the first parallel algorithms for arbitrary matroid intersections.

The algorithms in this paper are designed for a CRCW PRAM model of computation similar to the NYU Ultracomputer [7]. Synchronization in the NYU Ultracomputer is achieved using the *fetch-and-add* primitive. Our model also uses two *fetch-and-compare* primitives, *write-max* and *write-min*, which we believe are natural primitives for solving problems involving weights.

The next section of this paper contains definitions, a discussion of our *fetch-and-compare* primitives, and a brief description of the sequential algorithms upon which the parallel algorithms are based. Section 3 contains parallel algorithms for the unweighted and weighted two matroid intersection problems. Both algorithms use m^2 processors and have parallel time complexity $O(R(R + c(m)))$, where m is the number of elements in the matroids, R is the minimum of the ranks of the two matroids, and c(m) is the time to test for independence. The numbers of processors used by the algorithms are then reduced to obtain algorithms with linear speed-up. Section 4 shows how to decrease the parallel times to $O(R(\log R + c(m)))$ (using more processors). The last section contains a table summarizing our results.

This research was supported by Simon Fraser University President's Research Grant No. 02-4007 and by the Natural Sciences and Engineering Research Council of Canada under Grant No. A-0322.

2. PRELIMINARIES

Definition 1. A *matroid* $M = (E, \mathbf{I})$ consists of a finite set, E, of *elements* and a family, **I**, of subsets of E such that

1. $\varnothing \in \mathbf{I}$
2. If $X \in \mathbf{I}$ and $Y \subseteq X$ then $Y \in \mathbf{I}$
3. If $U, V \in \mathbf{I}$ with $|U| = |V| + 1$ then there exists an $x \in U - V$ such that $V \cup \{x\} \in \mathbf{I}$.

Subsets in **I** are called *independent sets.* All other subsets of E are *dependent sets.* A *base* of M is a maximal independent subset of E. A *circuit* of M is a minimal dependent subset of E. The *rank* of any subset $A \subseteq E$, denoted ρA, is the cardinality of a maximal independent subset of A. The rank of M, denoted ρM, is the rank of E. A *span* of $A \subseteq E$ is a maximal superset S of A satisfying $\rho S = \rho A$.

Definition 2. Let E be a finite set with a nonnegative weighting function $w : E \rightarrow \mathbf{R}^{+}$, and let $M_1 = (E, \mathbf{I}_1)$ and $M_2 = (E, \mathbf{I}_2)$ be two matroids over E. The *weighted two matroid intersection problem* is the problem of finding a set $I \in \mathbf{I}_1 \cap \mathbf{I}_2$ of maximum weight. The special case of $w(e) = 1$ for all $e \in E$ is the *cardinality two matroid intersection problem.*

The parallel algorithms in this paper are derived from serial algorithms by Lawler [9]. Lawler's algorithms are based on *augmenting sequences.* Let I be any intersection of two matroids M_1 and M_2. An *alternating sequence*, $e_1, e_2, e_3, \cdots$, with respect to I is constructed as follows. The first element e_1 is such that $I+e_1$ is independent in M_1. If $I+e_1$ is also independent in M_2, then the sequence is an *augmenting sequence.* Otherwise $I+e_1$ contains a unique circuit in M_2 and an element $e_2 \neq e_1$ in that circuit is chosen. $I+e_1-e_2$ is clearly independent in both M_1 and M_2. e_3 is an element for which $I+e_1-e_2+e_3$ is independent in M_1, whereas $I+e_3$ is not. If $I+e_1-e_2+e_3$ is independent in M_2, then the sequence is an augmenting sequence. Otherwise, this process continues until either an augmenting sequence if found or the set of elements is exhausted.

To facilitate the detection of circuits and the search for an augmenting sequence, a directed bipartite graph, BG(I), called the *border graph* of I, will be built. The two sets of vertices of BG(I) correspond to the sets $E-I$ and I. If element $e_i \in E-I$ forms a circuit with I in M_1, there are arcs (e_j, e_i) going from every element $e_j \in I$ in this circuit to e_i. If element $e_k \in E-I$ forms a circuit with I in M_2, then there are arcs (e_k, e_j) going from e_k to every element $e_j \in I$ in this circuit. Vertices in $E-I$ with indegree 0 are called *sources* and vertices in $E-I$ with outdegree 0 are called *sinks.* For the unweighted matroid intersection problem, it can be shown that the search for an augmenting sequence for I is equivalent to the search for a source-sink path without shortcuts in BG(I). A breadth-first search of BG(I) can be used to find such a path [9]. If an augmenting sequence is found, I is augmented by adding the "odd" elements to I and removing the "even" elements to yield a larger intersection, and the search for a new augmenting sequence begins. Otherwise, the algorithm terminates because an intersection is of maximum cardinality if and only if it admits no augmenting sequence. The augmenting path algorithm repeats the following steps until no source-sink path is found in BG(I).

(1) build the border graph BG(I) where I is the intersection currently being augmented
(2) perform a breadth-first search of BG(I) to find a source-sink path without shortcuts
(3) backtrack through the source-sink path augmenting I in the process

The serial algorithm for the weighted intersection problem is similar. The main difference is that an arbitrary source-sink path with no shortcut may not give a new intersection with maximum weight. An exhaustive search for all source-sink paths is needed to find an augmenting sequence of maximum incremental weight. Since the incremental weights for consecutive augmentations are non-increasing [9], the algorithm stops when an augmentation that gives a non-positive incremental weight is found.

Lawler [9] claims time complexities of $O(mR^2c(m)+m^2R)$ and $O(mR^2(m+c(m)))$ for his unweighted and weighted algorithms respectively. More careful analyses give $O(mR^2c(m))$ and $O(mR^2(R+c(m)))$ respectively, where m is the number of elements in the matroids, R is the minimum of the ranks of the matroids, and $c(m)$ is the maximum of the times to check for independence in the matroids. In both algorithms, building a border graph takes $O(mRc(m))$, backtracking and augmentation is $O(R)$, and there are at most R augmentations. The breadth-first search is $O(mR)$ in the unweighted algorithm and $O(mR^2)$ in the weighted algorithm.

We have designed the parallel algorithms in this paper for a CRCW (concurrent-read-concurrent-write) PRAM model of computation similar to the NYU Ultracomputer [7]. The model has p identical and independent processors and a global memory to which all processors have access. Concurrent reads and writes are allowed and are assumed to take constant time. When two or more processors write concurrently to the same memory location, an arbitrary processor succeeds. It is also assumed that all processors can be synchronized in constant time. Synchronization in the NYU Ultracomputer is achieved using the *fetch-and-add* primitive. Our model also uses two *fetch-and-compare* primitives, *write-max* and *write-min*. The implementation of the NYU Ultracomputer described in [7] uses an enhanced omega network. It is not difficult to modify the network and the network interface units to implement the *write-max* and *write-min* instructions. In fact, the hardware for these new operations is somewhat simpler than the hardware needed to implement *fetch-and-add* because no value is returned. The fetch-and-compare primitives can, of course, be efficiently simulated on any standard model, but we believe that they are natural primitives for solving problems involving weights.

Definition 3. If k processors $P_1, P_2, ..., P_k$ simultaneously perform instructions write-max(A, e_1), write-max(A, e_2),... write-max(A, e_k) respectively, then address A will be assigned the value of max{ $e_1, e_2, \cdots e_k$, C(A) } where C(A) is the original content of A. No value is returned to the processors. write-min(A, e) is defined similarly.

Some concurrency of *write-max* and *write-min* with other operations that access memory (*read, write*, and *fetch-and-add*) is possible. However, for simplicity we make all such concurrency illegal in our model.

3. PARALLEL MATROID INTERSECTION ALGORITHMS

In this section, we present parallel algorithms for the cardinality and weighted two-matroid intersection problems. The cardinality algorithm is, essentially, a direct translation of the serial algorithm and will only be outlined. The weighted algorithm is more complicated and will be discussed in more detail. Both algorithms use m^2 processors, P_{ij}, $1 \le i, j \le m$, one for each possible link in BG(I), the border graph for the current intersection I.

The algorithms consist of four steps which are executed once for each augmentation. Step 1 is initialization and takes constant time. Step 2 is the same for the two algorithms. It builds BG(I) and takes $O(c(m))$ time, where $c(m)$ is the maximum time to test for independence in either matroid. Step 4 is the augmentation step. The actual backtracking and augmentation loop is the same in both algorithms; only the stopping conditions for the algorithms differ. Since a shortest source-sink path in BG(I) has length at most $2R$ in both cases, Step 4 is $O(R)$ in both algorithms.

The main differences between the algorithms are in the breadth-first search in Step 3. In the cardinality algorithm, this is done by growing a forest of directed trees with the sources as the roots and the links in BG(I) as the branches. The loop in Step 3 is synchronized, so the forest grows by one level during each iteration and the nodes on this level are "active". The sources are the "active" nodes initially. In each iteration, each active node, e_i, examines all its outgoing edges. If a son, e_j, is found which has no assigned parent, then, e_i will attempt to make itself the parent of e_j in the forest. Several active nodes may simultaneously attempt to be the parent of the same node, e_j. In our parallel model, one processor will succeed and it is not important which one succeeds; any search path to e_j at this point will have the same length. If e_j is a sink, then a source-sink path is found. Otherwise, e_j will be an active node during the next iteration. Since the length of the shortest source-sink path is at most $2R$, and each level of this search is done in constant parallel time, Step 3 is $O(R)$. Since there are at most R augmentations, the total time is $O(R(R + c(m)))$.

In the weighted algorithm, Step 3 will find a source-sink path with the greatest incremental weight if it exists. This path will correspond to a maximum weight augmenting sequence. As in the cardinality algorithm, a forest of directed trees is grown one level at a time. Before each new level is grown, each element e_j is assigned a local weight WT_j. WT_j is the given weight $w(e_j)$ if $e_j \notin I$ and $-w(e_j)$ if $e_j \in I$. For each element e_i, a variable Δ_i records the greatest incremental weight among all paths from a source to e_i that have been searched so far. At each level, each active node e_i examines all of its outgoing edges in BG(I) and attempts to make each of the corresponding nodes its son (including nodes that already have a father). An active e_i for which $\Delta_i + WT_j$ is a maximum becomes the new parent of e_j if $\Delta_i + WT_j$ is also greater than the current Δj, and $\Delta_i + WT_j$ is recorded in Δ_j using the write-max instruction at line 5. If e_j is a sink then a potential augmenting sequence has been found. The write-max at line 7 will retain only the source-sink path with the maximum incremental weight. The sink of the chosen path will be entered into AUG. If e_j is not a sink then it will be activate in the next iteration. The search will continue until all paths from the sources are exhausted. Except for the first iteration, a node will be "active" if and only if it is not a sink and it has been assigned a "new" parent in the previous iteration. Note that an element may be assigned a different parent during each iteration. All nodes e_i activated in the previous iteration are deactivated at line 6. However, a node can be activated again at line 8 in the same iteration. Hence, a node may be activated in more than one iteration of the search.

The proof that Step 3 will find a source-sink path with maximal incremental weight in BG(I) is now fairly mechanical and is omitted. The incremental weights for consecutive augmentations in this algorithm must be decreasing as they are in the serial algorithm, so Step 4 correctly stops the computation when it discovers a non-positive incremental weight Δ_s. The depth of the search in Step 3 is the length of the maximum weight augmenting sequence and this is no longer than $2R$. Each level of this search takes constant time, so the total time for Step 3 is $O(R)$. There are at most R augmentations, so the overall running time of the algorithm is $O(R\,(R+c(m)))$.

The parallel weighted matroid intersection algorithm is shown below. The symbol '$\leftarrow *$' means concurrent write, and the symbol '$\leftarrow **$' means concurrent write of possibly different values to the same location. *<synchronize>* indicates a synchronization point. No processor can proceed past a synchronization point until all processors have reached that point. The algorithm uses the following variables.

Local Variables

$WEIGHT(i)$	the given weight $w(e_i)$ for element e_i
WT_i	weight contributed by element e_i to the incremental weight of a source-sink path through e_i
$LINK_{ij}$	true if $(e_i, e_j) \in$ BG(I)

Global Variables

I	boolean vector indicating membership of elements in the current intersection
$SOURCE_j$	true if e_j is a source in BG(I)
$SINK_j$	true if e_j is a sink in BG(I)
$ACTIVE_j$	true if e_j is "active"
$PARENT_j$	i if e_i is the parent of e_j in the breadth-first forest
Δ_i	greatest incremental weight among all paths from a source to e_i
Δ_s	weight of the augmenting sequence
$ENDSEARCH$	true if the search for source-sink path is done
AUG	current node on source-sink path during backtracking

Algorithm 1. Parallel Weighted Matroid Intersection

$I = \emptyset$ initially.
Each processor P_{ij}, $1 \le i, j \le m$ executes the following steps.

Step 1 : (* *Initialization* *)

```
1.  $LINK_{ij} \leftarrow$ false;  $ACTIVE_j \leftarrow *$ false;  $PARENT_j \leftarrow *$ null;  $AUG \leftarrow *$ null
    $LINK_{ii} \leftarrow$ true;  $\Delta_s \leftarrow * \ -\infty$
2.  $SOURCE_j \leftarrow *$ true;  $SINK_j \leftarrow *$ true
    if $e_j \in I$ then
        $SOURCE_j \leftarrow *$ false ;  $SINK_j \leftarrow *$ false
```

Step 2 : (* *Building Border Graph BG(I)* *)

```
1.  if $e_i \in I$ and $I + e_j \notin I_1$ and $I + e_j - e_i \in I_1$ then
        $LINK_{ij} \leftarrow$ true;  $SOURCE_j \leftarrow *$ false
2.  if $e_j \in I$ and $I + e_i \notin I_2$ and $I + e_i - e_j \in I_2$ then
        $LINK_{ij} \leftarrow *$ true;  $SINK_i \leftarrow *$ false
```

```
Step 3 : (* search for source-sink path *)
if $e_j \notin I$ then $WT_j \leftarrow^* WEIGHT(j)$ else $WT_j \leftarrow^* -WEIGHT(j)$
<synchronize>
ENDSEARCH $\leftarrow^*$ true
if $SOURCE_j$ then
        $\Delta_j \leftarrow WT_j$;   $ACTIVE_j \leftarrow^*$ true;        ENDSEARCH $\leftarrow^*$ false
while not ENDSEARCH
<synchronize>
        ENDSEARCH $\leftarrow^*$ true
        if $ACTIVE_i$ and $LINK_{ij}$ then
                write-max($\Delta_j, \Delta_i + WT_j$)
        <synchronize>
                $ACTIVE_i \leftarrow^*$ false
                if $\Delta_j = \Delta_i + WT_j$ then $PARENT_j \leftarrow^{**} i$
                if $SINK_j$ then
                        write-max($\Delta_s, \Delta_j$)
                <synchronize>
                        if $\Delta_s = \Delta_j$ then $AUG \leftarrow^{**} j$
                else
                        ENDSEARCH $\leftarrow^*$ false;  $ACTIVE_j \leftarrow^*$ true
        <synchronize>
end-while

Step 4 : (* augmentation *)
if $\Delta_s = -\infty$ then
        $I$ is both maximum cardinality and maximum weight, STOP
else
        if $\Delta_s \le 0$ then $I$ is of maximum weight, STOP
if $SOURCE_{AUG}$ then $I \leftarrow I + e_{AUG}$
else
while augmentation not done
        if $i = PARENT_j$ and $j = AUG$ then
                if $SOURCE_i$ then $I \leftarrow^* I + e_i$   (* augmentation is done *)
                else
                        if $e_j \in I$ then $I \leftarrow^* I - e_j$ else $I \leftarrow^* I + e_j$
                        $AUG \leftarrow^* i$
<synchronize>
end-while
go to Step 1
```

The time-processor product for both parallel algorithms is $O(m^2R(R+c(m)))$. This is greater than the time bounds for the two serial algorithms. However, Brent's theorem can be used to reduce the number of processors to achieve linear speed-up (see [11]).

Theorem 1 (Brent). Any synchronized parallel algorithm of depth d that consists of a total of x elementary operations can be implemented by p processors within a depth of $\lceil x/p \rceil + d$.

Theorem 2. The cardinality two matroid intersection problem can be solved in parallel time $O(R(R+c(m)))$ with $\min(mc(m), Rm)$ processors.

Proof. It is not hard to show that the number of elementary operations in the parallel cardinality algorithm is $O(mR^2c(m))$, the same as the serial algorithm from which it is derived. The depth has been shown to be $O(R(R+c(m)))$. By Theorem 1, the algorithm can be implemented using p processors within a depth of $\lceil O(mR^2c(m))/p \rceil + O(R(R+c(m))$. $\min(mc(m), Rm)$ processors gives depth of $O(R(R+c(m))$ and a time-processor product of $O(mR^2c(m))$ which is the same as the time of the sequential algorithm. □

Theorem 3. The weighted two matroid intersection problem can be solved in parallel time $O(R(R+c(m)))$ with mR processors.

Proof. The number of elementary operations in the parallel weighted algorithm is $O(mR^2c(m)+mR^3)$, the same as the serial algorithm. The depth has been shown to be $O(R(R+c(m)))$, so by Theorem 1, the algorithm can be implemented in a depth of $O(R(R+c(m))$ with mR processors. The time-processor product is $O(mR^2c(m)+mR^3)$, the same as the time of the sequential algorithm, so a linear speed-up is achieved. □

4. REDUCING THE DEPTH

Both parallel algorithms in the previous section have complexity $O(R(R+c(m)))$. The depth of both algorithms can be reduced by reducing Steps 3 and 4 from $O(R)$ to $O(\log R)$. The backtracking in Step 4 can easily be accomplished in $O(\log R)$ time with m^2 processors by a routine application of *recursive doubling* (see [2]). We show here how to reduce the depth of the breadth-first search in Step 3 to $O(\log R)$ by regarding the problem as a shortest path tree problem. The method is an adaptation of a graph-based version of a repeated min-plus multiplication algorithm (see [3] and [10]).

Initially, a tree T_i is built for each element e_i in E. The tree for e_i will have e_i at its root (level 0) and the sons of e_i in $BG(I)$ as the leaves at level 1. In the next stage, each leaf e_j at level 1 of each tree is examined and an attempt is made to hook the tree T_j rooted at e_j onto this leaf. The height of the trees becomes 2. In the third stage, each leaf e_k at level 2 of each tree is examined and an attempt is made to hook the tree rooted at e_k onto this leaf. In general, at the q-th stage, the trees will have height 2^q and the leaves at the 2^q-th level are examined. This is repeated until the search is over.

In the trees generated by this search, each node will correspond to an element in the matroids (i.e., a node in the border graph $BG(I)$). Consider the tree T_i that would result at the q- th stage if, at each of the earlier stages (stages 1 to q-1), the whole tree T_j for each leaf e_j is hooked to T_i. This tree could have many nodes representing the same element in the matroid. However, in a breadth-first search tree, we want at most one node at a lowest possible level (closest to the root) to be retained for each element. So, when we link a tree T_j to a leaf of T_i, we usually do not want to add the whole tree T_j. We check every node in T_j to see if the element it represents already exists in T_i. Also, since the hooking of trees at the leaves is done in parallel, we check if some other tree is adding a node for the same element at another leaf of T_i. We do not add a node if another node is or will be in the tree T_i at a lower level (closer to the root). The result is that each element has at most one node in each tree and the size of each tree is at most m.

The breadth-first search for the weighted algorithm uses a method similar to the unweighted case. Here we need a variable $\Delta(i,j)$ to store the greatest incremental weight among all paths from element e_i to element e_j which have been searched so far. These Δ values are attached to the nodes in the trees. As in the unweighted algorithm, we want to retain at most one node corresponding to each matroid element. However, in this case it is a node with the greatest $\Delta(i,j)$ value, rather than a node at the lowest level. A *write-min* instruction is used in the unweighted case to ensure that the node with lowest level succeeds, while a *write-max* instruction is used (at line 8 in the algorithm below) to ensure that the node with greatest value succeeds in the weighted case. If there is more than one node with greatest $\Delta(i,j)$ value in the weighted case, then a *write-min* instruction (at line 9 in the algorithm below) is used to choose the one with the lowest level. Conflicts among sinks are resolved similarly at lines 11 and 12 in the algorithm below as are conflicts among source-sink paths at lines 14 and 15. The unweighted algorithm can stop as soon as it finds a source-sink path. The weighted algorithm must exhaust all source-sink paths to determine the one with maximum incremental weight, so a tree will remain active until no element exists at the 2^{q+1} level.

The following algorithm implements this breadth-first search strategy for the weighted case. It replaces Step 3 of Algorithm 1 described in the previous section. The algorithm uses the following variables.

Local Variables

$WEIGHT(i)$	the given weight $w(e_i)$ for element e_i
$LINK_{ij}$	true if $(e_i, e_j) \in \mathrm{BG}(I)$

Global Variables

I	boolean vector indicating membership of elements in the current intersection
$SOURCE_j$	true if e_j is a source in $\mathrm{BG}(I)$
$SINK_j$	true if e_j is a sink in $\mathrm{BG}(I)$
T_i	tree with e_i as its root.
$AUGT$	the tree that contains the augmenting path.
AUG	the sink in the augmenting path.
Δ_{fs}	the incremental weight of the augmenting path.

Each element $e_j \in E$ may appear at most at one node in T_i. If e_j is in T_i then it has

(1) $\mathrm{PARENT}(i,j)$ = index number of its parent in T_i

(2) $\mathrm{LEVEL}(i,j)$ = the level number of e_j in T_i (The root has level number 0.)

(3) $\Delta(i,j)$ = temporary incremental weight of path from e_i to e_j

If e_j is not in T_i then $\mathrm{PARENT}(i,j)$=null and $\Delta(i,j)=-\infty$.

Tree T_i also has two variables:

(1) $\mathrm{SINK}(T_i)$ = the sink with greatest $\Delta(i,j)$ among all sinks in T_i

(2) $\mathrm{SINKLEVEL}(T_i)$ = the level of $\mathrm{SINK}(T_i)$

(3) $\Delta_s(T_i)$ = incremental weight from i to $\mathrm{SINK}(T_i)$.

If T_i does not contain a sink then $\mathrm{SINK}(T_i)$ is null, $\mathrm{SINKLEVEL}(T_i)$ is ∞, and $\Delta_s(T_i)$ is $-\infty$.

Algorithm 2. Weighted Breadth-First Search

Each processor P_{ijk}, $1 \le i,j,k \le m$ executes the following steps.

Initialization :

```
1.  for each element $e_i \in E$
2.      (* build tree $T_i$ with $i$ as its root *)
        (* initially $\mathrm{SINK}(T_i)$ = null, $\mathrm{SINKLEVEL}(T_i) = \infty$, $\Delta_s(T_i) = -\infty$ *)
        $\mathrm{PARENT}(i,i) \leftarrow^* 0$;  $\mathrm{LEVEL}(i,i) \leftarrow^* 0$;  $\Delta(i,j) \leftarrow^* -\infty$
        if $i \in I$ then $\Delta(i,i) \leftarrow^*$ -WEIGHT(i) else $\Delta(i,i) \leftarrow^*$ WEIGHT(i)
3.      if $SINK_i$ then
                $\Delta_s(T_i) \leftarrow^* \Delta(i,i)$;   $\mathrm{SINK}(T_i) \leftarrow^* i$;   $\mathrm{SINKLEVEL}(T_i) \leftarrow^* 0$
4.      for each element $e_j \neq e_i$
            if $LINK_{ij}$ then
                $\mathrm{PARENT}(i,j) \leftarrow^* i$;  $\mathrm{LEVEL}(i,j) \leftarrow^* 1$
5.          if $j \in I$ then $\Delta(i,j) \leftarrow^* WEIGHT(i) - WEIGHT(j)$
                          else $\Delta(i,j) \leftarrow^* WEIGHT(j) - WEIGHT(i)$
6.          if $e_j$ is a sink and $\Delta(i,j) > \Delta_s(T_i)$ then
                write-max( $\Delta_s(T_i), \Delta(i,j)$ )
            <synchronize>
                if $\Delta_s(Ti) = \Delta(i,j)$ then
                    $\mathrm{SINK}(T_i) \leftarrow^* j$;  $\mathrm{SINKLEVEL}(T_i) \leftarrow^* 1$
        if no element appears at level 2 then $T_i$ is inactive

    q ← -1
```

```
repeat the following until DONE
q-th iteration :
$q \leftarrow q + 1$
7.  for each active $T_i$
        for each $e_j$ at level LEVEL$(i,j) = 2^q$
            (* $T_j$ is examined *)
8.          for each $e_k$ in $T_j$
                write-max($\Delta(i,k)$, $\Delta(i,j) + \Delta(j,k)$)
            <synchronize>
9.              if $\Delta(i,k) = \Delta(i,j) + \Delta(j,k)$ then
                    write-min( LEVEL$(i,k)$, LEVEL$(i,j) + 2^q$)
                <synchronize>
10.                 if LEVEL$(i,k)$ = LEVEL$(i,j) + 2^q$ then
                        PARENT$(i,k) \leftarrow$** PARENT$(j,k)$
11.             if SINK$(T_j) \neq$ null then
                    write-max($\Delta_s(T_i)$, $\Delta_s(T_j)+\Delta(i,j)$ )
                <synchronize>
12.                 if $\Delta_s(T_i)=\Delta_s(T_j)$ then
                        write-min( SINKLEVEL$(T_i)$, SINKLEVEL$(T_j) + 2^q$ )
                    <synchronize>
13.                     if SINKLEVEL$(T_i)$ = SINKLEVEL$(T_j) + 2^q$ then
                            SINK$(T_i) \leftarrow$** SINK$(T_j)$
14.     if $e_i$ is a source and SINK$(T_i) \neq$ null then
            write-max( $\Delta_{fs}$, $\Delta_s(T_i)$ )
        <synchronize>
15.         if $\Delta_{fs}=\Delta_s(T_i)$ then write-min( SL, SINKLEVEL$(T_i)$ )
            <synchronize>
16.             if SL = SINKLEVEL$(T_i)$ then $AUGT \leftarrow$** $T_i$
                <synchronize>
17.                 if $AUGT = i$ then $AUG \leftarrow$ SINK$(T_i)$
        if no element exists at level $2^{q+1}$ then
            $T_i$ becomes inactive
    if there is no active tree then DONE is true.
```

Four technical lemmas are needed to prove the correctness of the two breadth-first search algorithms. The proofs are relatively straightforward and are omitted.

Lemma 4. A node in T_j will not be added to a tree T_i unless all of its ancestors in T_j are added to T_i.

Lemma 5. At the q-th iteration, no new nodes can be added as descendents to leaves at levels lower (closer to the root) than level 2^q.

Lemma 6. A tree which has no node at the 2^{q+1}-th level after the q-th iteration will not acquire new nodes during later iterations even if it remains active.

Lemma 7. Step 3 is a breadth-first search of the border graph.

Theorem 8. The search for a source-sink path can be done in $O(\log R)$ time using m^3 processors.

Proof. The heights of the trees grown by the breadth-first search double each iteration, and the maximum length of an augmenting path is $2R$, so the number of iterations is $O(\log R)$. Each iteration can be done in constant time as follows. At the 0-th iteration of both algorithms, we can assign a processor to each of the m possible links in each of the m trees. Hence m^2 processors are required for this iteration. At the q-th (q>0) iteration of both algorithms, we can assign processor PE_{ijk} to handle each active tree T_i, each element e_j at level 2^q of T_i, and each e_k in T_j. There are m possible active trees, each tree has less than m possible leaves at level 2^q and each tree has m possible elements, so m^3 processors will be needed. ☐

Theorem 9. For the cardinality problem, the search for a source-sink path can be done in $O(\log R)$ time using $m^3/\log R$ processors.

Proof. For the cardinality algorithm, the required number of processors as stated in Theorem 8 can be reduced by using Brent' s Theorem. In this algorithm, each node can become a leaf at the 2^q level for at most one q value because it cannot be added twice to a tree. Hence the total number of leaves at the 2^q level for all q is no more than m. Since each such leaf requires $O(m)$ operations when a tree is linked to it, and there are m trees in total, the number of elementary operations is $O(m^3)$. Applying Brent's Theorem, we need $m^3/\log R$ processors to achieve the same time bound.

Theorem 10. The cardinality two-matroid intersection problem can be solved in $O(R(\log R + c(m)))$ time using $O(m^3/\log R)$ processors.

Theorem 11. The weighted two-matroid intersection problem can be solved in $O(R(\log R + c(m)))$ time using $O(m^3)$ processors.

5. SUMMARY

We have presented $O(R(\log R + c(m)))$ time parallel algorithms for both the weighted and unweighted versions of the two matroid intersection problem. Our model is a CRCW PRAM with two new primitives, *write-max* and *write-min*. These operations can be simulated in $O(\log p)$ time on a standard CRCW PRAM with p processors. The times of our algorithms on different models are summarized below.

The Unweighted Two-Matroid Intersection Problem

Sequential time [9]: $O(mR^2c(m))$

Parallel time:

(i)	$O(R(\log R + c(m)))$	if $p \geq m^3/\log R$ using *write-min*
(ii)	$O(R(R + c(m)))$	if $p \geq \min(mc(m), mR)$ not using *write-max* or *write-min*
(iii)	$O(\frac{mR^2c(m)}{p})$	if $p \leq \min(mc(m), mR)$ not using *write-max* or *write-min*

The Weighted Two-Matroid Intersection Problem

Sequential time [9]: $O(mR^2c(m) + mR^3)$

Parallel time:

(i)	$O(R(\log R + c(m)))$	if $p \geq m^3$ using *write-max* and *write-min*
(ii)	$O(R(R + c(m)))$	if $p \geq mR$ using *write-max*
(iii)	$O(\frac{mR^2c(m) + mR^3}{p})$	if $p \leq mR$ using *write-max*
(iv)	$O(R^2\log m + Rc(m))$	if $p \geq mR$ not using *write-max* or *write-min*
(v)	$O(\frac{mR^2c(m) + mR^3\log m}{p})$	if $p \leq mR$ not using *write-max* or *write-min*

REFERENCES

[1] Allan Borodin, Joachim von zur Gathen, and John Hopcroft. "Fast Parallel Matrix and GCD Computation." *Proc. 23rd Symposium on Foundations of Computer Science*, 1982, pp. 65-71.

[2] Stephen A. Cook. "The Classification of Problems which have Fast Parallel Algorithms." *Technical Report, No. 164/83, Dept. of Computer Science, University of Toronto.*

[3] E. Dekel, D. Nassimi and S. Sahni. "Parallel Matrix and Graph Algorithms." *SIAM J. Comput. 10:4* (Nov. 1981), pp. 657-675.

[4] E. Dekel and S. Sahni. "A Parallel Matching Algorithm for Convex Bipartite Graphs." *Proc. 1982 International Conference on Parallel Processing*, pp. 178-184.

[5] Ada Fu and Joseph G. Peters. "Parallel Bipartite Matching Algorithms." *Unpublished.*

[6] Michael R. Garey and David S. Johnson. "Computers and Intractability : A Guide to the Theory of NP-Completeness." *W.H. Freeman and Company,* 1979.

[7] Allan Gottlieb, Ralph Grishman, Clyde P. Kruskal, Kevin P. McAuliffe, Larry Rudolph, and Marc Snir. "The NYU Ultracomputer : Designing an MIMD Shared Memory Parallel Computer." *IEEE Trans. on Computers, Vol C-32* (Feb 1983), pp. 175-189.

[8] S.C. Kwan and W.L. Ruzzo. "Adaptive Parallel Algorithms for Finding Minimum Spanning Trees." *Proc. 1984 International Conference on Parallel Processing*

[9] Eugene L. Lawler. "Combinational Optimization : Networks and Matroids." *Holt, Rinehart and Winston,* New York, 1976.

[10] Michael J. Quinn and Narsingh Deo. "Parallel Graph Algorithms." *Computing Surveys* 16 : 3, (Sept. 1984), pp. 319-348.

[11] Yossi Shiloach and Uzi Vishkin. "An $O(n^2 \log n)$ Parallel MAX-FLOW Algorithm." *Journal of Algorithms 3,* (1982), pp. 128-146.

SOLVING COMBINATORIAL PROBLEMS ON ARRAYS WITH ONE-WAY DATAFLOW (Extended Abstract)

CARLA D. SAVAGE, MATTHIAS STALLMANN, JO ELLEN PERRY
Department of Computer Science, North Carolina State University
Raleigh, North Carolina 27695-8206

Abstract.

In this paper, we present some algorithms for solving combinatorial problems on one-dimensional processor arrays in which data flows in only one direction through the array. The problems we consider are: ranking the elements in a chain of size n, rooting a spanning tree with n vertices, and computing biconnected components of a connected graph with n vertices. We show that each of these problems can be solved using arrays of size n in which the data enters at the first cell and flows through the array in only one direction until it leaves the last cell as output. We show also how the biconnectivity algorithm for the array yields a new sequential algorithm for computing biconnected components.

1. INTRODUCTION

In [2] it was shown that the problems of ranking the elements in a chain of size n and rooting a spanning tree with n vertices could be solved on a two-dimensional square mesh of n cells in time $O(\sqrt{n})$. By standard techniques [3], these algorithms can be transformed to solve the same problems on an array of $O(n)$ cells in time $O(n)$. However, the solutions of [2] are recursive and use four-way data flow so that the resulting one-dimensional array solution would be recursive and use two-way data flow. In this paper, we use ideas from [2] together with new techniques to solve the problems of ranking a chain and rooting a spanning tree on a one-dimensional array with one-way data flow, yet with the same computational resources. Our solutions have the property that input data enters the array in arbitrary order and at regular time intervals and that successive instances of the problem can be pipelined.

Algorithms for determining connected components and finding a spanning tree of an undirected n-vertex graph on an array of n cells with one-way data flow were presented in [4] and [12]. Given a rooted spanning tree of a graph G, an algorithm was presented in [1] to determine the biconnected components of G on a one-dimensional array of size n using only one-way data flow. In [1], however, computing a spanning tree required $\log n$ passes of the data through an array of size n. Combining our new results with previous results, we can compute biconnectivity information using a sequence of c arrays of size n, where c is some small constant, such that the edges of the graph enter the first array as input and succeeding arrays are fed by the output of their predecessor.

On a two-dimensional square mesh, there are algorithms to determine biconnected components with n^2 cells in time $O(n)$ [3] or with m cells in time $O(\sqrt{m})$ [2], where m is the number of edges in the graph. Our one-dimensional array algorithm uses only $O(n)$ processors to determine whether the graph is biconnected. It appears to be necessary, however, to recycle the edges of the graph through the array. This requires $O(m)$ storage area for the edges waiting to be recycled. So, although the one-dimensional array algorithm takes the same asymptotic amount of area as the mesh in [2], the number of processors will be much smaller for dense graphs.

Finally, we present a new biconnectivity algorithm for one-way one-dimensional arrays. Although the biconnectivity algorithm presented in [1] was relatively simple, the proof that it works was not. We modified the algorithm of [1] in such a way that we could get a meaningful proof of its correctness. The resulting algorithm has two sequential implementations that are of interest. One uses the set merging results of [6] to get another linear time biconnectivity algorithm. The other implementation uses set merging to compute the initial spanning tree as well as the biconnected components. The expected running time is linear and only $O(n)$ locations of random access memory are required.

In Section 2, we present a model of computation and some conventions for describing algorithms on the model. In Section 3 we discuss the chain rank algorithm and its application to the Euler Tour technique for rooting a spanning tree. In Section 4, we present an algorithm for finding biconnected components on a one-dimensional array, and we discuss sequential implementations of the algorithm.

2. MODEL OF COMPUTATION

The model of computation we use is a linear array of n cells. Each cell has a small constant number of data registers and a processor capable of executing a simple program in constant time independent of the length of th input. We assume each cell contains the same program and that the array operates synchronously in a sequence of steps called "time units." During a time unit, each cell i takes as input the output values of cell $i-1$ and executes its program on its current input and the values in the data registers, causing the data registers to be updated and output values to be generated. Cell 1 takes as input the value waiting to enter the array and the output of the array at time t is the output generated by cell n at time $t-1$. Data flow through the array, therefore, is one-way.

We assume that for a given problem instance, the sequence of data to be input into the array is demarcated by endmarkers which can be used to turn on and turn off program execution in the cells. That is, if a cell gets the "start" marker as input at time t, it will begin to execute its program at time $t+1$ and continue to do so until it receives the "stop" marker.

Our main interest in studying one-way arrays is that algorithms on one-way arrays can be made to have nice properties with respect to problem decomposition and fault tolerance ([8],[10]). If certain conditions are satisfied by a one-way linear array algorithm, a problem which ideally would require an array of length n could be solved on an array of size $k < n$ simply by recycling the data through the array $\lceil n/k \rceil$ times. If cells were designed so that in case of cell failure, inputs could be passed through to output ports unchanged, then an array of size $n+j$ could tolerate the failure of any j cells (excluding possibly cells 1 and n) and still work correctly. Further, an array with at least k functioning cells can solve the problem with no more than $\lceil n/k \rceil$ passes of the data through the array.

In order to have these properties of decomposability and fault tolerance, the following property must be satisfied [10]. The output of a cell must depend only on the program in the cell and the data which has so far passed through the cells. In particular, data registers in a cell are not initialized before the "start" marker enters. All information is in the data stream and instructions for interpreting the information are within the program in the cell. An increase in the delay between two successive inputs entering a cell should not alter the output values produced by the cell.

Systolic array architectures have often been criticized because of the need for a global clock. An additional advantage of one-way arrays having the properties described above is that global synchronization is not required. Each cell begins its next time unit when new data is available from the cell on the left and ends a time unit by signalling the cell on the right, telling it that new data is available. We will continue to refer to the computation in terms of distinct time units even though not all cells are executing the same time unit simultaneously.

In [10] we show that a one-way algorithm which requires the data registers in cells to be initialized can be transformed into one in which the initial state information is included in the data stream. The penalty is an increase in the length of the input by an additive term of size $O(n)$ and a small (constant) increase in the size and time of the program in each cell to enable the additional data to be interpreted. Thus, there is no asymptotic penalty incurred, for example, in designing a graph algorithm for a linear array in which each cell is assumed to be preloaded with the vertex to which it corresponds (as in [4]). We will use this result to simplify the description of the algorithms in this paper.

The algorithms we present in this paper will assume that the data is "pushed" through the array. Informally, this means that a cell will not generate an output value until it is holding two data items. That is, during a time unit, an input value will not necessarily result in an output value. It may be held until a new input value is finally received, at which time an output will be generated. Local synchronization, as described above, can be used to inform the cell to the right when new output is available. The "stop" marker can be used to clear out computed values from the array. (If a cell receives the "stop" marker at time t, it outputs its internal state at time $t+1$ and the stop marker at time $t+2$.) The mechanism for effecting this push mode of data flow can be incorporated into the program within each cell and we assume in our algorithms that this is the case.

The program within a cell could actually consist of a small constant number of different programs, selected by special markers in the data stream that enter the cell as input. The mechanism for this is the same as that used for initializing data within each cell. Thus, although each cell is executing the same main program, different sections of the array could be executing different subprograms during a time unit.

All data flow is one-way through the array and the output of a cell depends only on its program and the data that has passed through the cell since the "start" marker was received. Thus everything - instructions, data, and successive problem instances - can be pipelined through the array. We can view an array algorithm as a loop in which each iteration reads one input data item and modifies the contents of the whole array in response to that item.

3. CHAIN RANKING AND ROOTING A TREE

A chain $C = (V,E)$ is a graph consisting of a single directed path. If s is the start vertex of the chain, the rank of a vertex x is the number of edges along the path from s to x. Among the applications for computing the rank of a chain is rooting a tree. If T is an acyclic, undirected, connected graph, rooting T involves selecting a vertex of T as root and orienting all the edges of T to point from a node to its parent.

The Euler Tour technique from [13] for rooting a tree is: (1) Make T Eulerian by replacing each edge $\{u,v\}$ by directed edges (u,v) and (v,u). (2) Choose a root r and rank edges along the Euler tour, starting at r. (3) Eliminate the lower ranked copy of each edge.

We modified the approach of [2] slightly to do step (1). Let L be a list of edges of T. Construct a simple directed cycle D as follows. The vertex set, $V(D)$, contains for each vertex v of T the vertices $v_0, v_1, \ldots, v_{deg(v)-1}$, where $deg(v)$ is the degree of v in T. For each edge $\{a,b\}$ of T, if $\{a,b\}$ is preceded on L by exactly p edges incident with a and q edges incident with b, then D contains the edges $(a_p, b_{(q+1)\bmod deg(b)})$ and $(b_q, a_{(p+1)\bmod deg(a)})$.

The construction of D can be implemented on a one-dimensional array of size $\lceil n/2 \rceil$ by "folding" [11]. Step (2) can be performed by chain ranking. Step (3) can be performed as a sort, modified to kill the first copy of each edge.

Given the edges of a chain C in arbitrary order, we compute the rank of each vertex in two phases. In phase 1, edges enter the array as chains of length 1. Chains move right through the array, merging whenever possible with contiguous subchains. As merging proceeds, a tree T is constructed: when chain $x \rightarrow y$ merges with $y \rightarrow z$, y and z become children of x in T. The weight of an edge (x,y) in T (y is the parent of x) is the distance from y to x on the chain. T is stored in the array in such a way that during phase 2, each vertex x moves to the right, tracing a path to the root of T (which is s). The rank of x is the sum of the weights along the path.

The algorithm performed by each cell during chain ranking is described on the next page. Each chain $x \rightarrow y$ is represented by a record of the form $(x,y,d,phase)$, where d is the distance from x to y and phase indicates which phase of the computation the record is involved in. Input records have the form $(x,y,1,1)$, where (x,y) is an edge, and output records have the form $(s,x,d,2)$, where d is the distance from s to x. The "start" marker informs cells of the value of n or the identity of the first and last vertices on the original chain. Two records are called *contiguous* if the chains they represent have a common endpoint.

During phase 1, chains move right, folding over each other until every vertex but the first and the last becomes the midpoint of exactly one merge operation. When $x \rightarrow y$ is merged with $y \rightarrow z$ to form $x \rightarrow z$, the record for $x \rightarrow y$ begins phase 2 as an edge of T. Because the phase 1 record for $x \rightarrow z$ moves right, a chain ending at x will begin phase 2 in a cell to the right of the one where $x \rightarrow y$ began phase 2. Thus, the edges of T begin phase 2 topologically sorted from the leaves to the root. More details of the correctness argument for the chain rank algorithm are given in [11].

Chain Ranking Algorithm.

```
new_R := INPUT
/* There are now two records in the cell, new_R, the most recent in-
   put, and R, the record stored by the previous computation. */
if new_R = $(s,t,n,1)$, where $s$ and $t$ are the ends of the chain then
        new_R := $(s,t,n,2)$ end if
if R = "start" then
        R := new_R; OUTPUT := "start"
else if new_R = "stop" then
        change state as required; OUTPUT := R
else if both records are in phase 1 and contiguous then
        let the records be $(x,y,d_1,1)$ and $(y,z,d_2,1)$
        R := $(x,y,d_1,2)$; OUTPUT := $(x,z,d_1+d_2,1)$
else if both records are in phase 2 and contiguous then
        let the records be $(x,y,d_1,2)$ and $(y,z,d_2,2)$
        /* By Lemma 3.4 we know R = $(x,y,d_1,2)$ and
           new_R = $(y,z,d_2,2)$ */
        R := $(x,y,d_1,2)$; OUTPUT := $(x,z,d_1+d_2,2)$
else if one record is in phase 1 and the other in phase 2 then
        R := the phase 2 record; OUTPUT := the phase 1 record
else
        OUTPUT := new_R
end if
```

4. BICONNECTIVITY PHASE

In this section we discuss how, given a rooted directed spanning tree T of a graph G, we can partition the edges of T into biconnected components wrt G. As a byproduct of our array algorithm, we also present an efficient sequential biconnectivity algorithm which is fundamentally different from previously published algorithms (e.g., [13]) and may have significant advantages in settings where page faults are a problem. Both algorithms can be seen as implementations of the same basic idea, described below.

Throughout this discussion, $p(v)$ denotes vertex v's parent in T. We identify vertices of G with edges of T in the following way: v corresponds to the unique edge from v to $p(v)$; the root of T corresponds to a dummy edge which plays no significant role in the algorithms. From now on, we can think of the vertices of G as being grouped into biconnected components. Specifically, u is in the same component as v iff the edge $\{u,p(u)\}$ is in the same component as $\{v,p(v)\}$. It is also convenient to think of the vertices as being in a topological ordering wrt T. We will assume that vertices are numbered in such a way that $p(v) > v$.

The algorithms work by repeatedly modifying the partition of $V(G)$ into equivalence classes (components). Initially, each vertex is in a class by itself. The equivalence classes of two vertices are merged whenever the corresponding edges are known to appear in the same fundamental cycle wrt T. The correctness of our approach is established by the fact that two edges of a graph are in the same biconnected component iff, with respect to some spanning tree, they lie in the same simple cycle which is the symmetric difference of a collection of fundamental cycles.

The final partition of $V(G)$ into equivalence classes may also be used to compute biconnected components on all of G (the edge $\{u,v\}$, where $u < v$, is in the same class as u), articulation points (v is an articulation point iff (i) v is the root of T and two of its children are in different classes or (ii) v is not the root of T and any child of v is in a different class from v), and bridges.

The basic algorithm can be described as follows. We denote the least common ancestor of x and y in T by $lca(x,y)$; $L(x,y)$ is a list of all vertices which are ancestors of either x or y but proper descendants of $lca(x,y)$ in T. Note that each iteration of the loop effectively puts all edges of the fundamental cycle in $T \bigcup \{\{x,y\}\}$ into the same class.

Basic Algorithm

```
Initialize each vertex to be in a set by itself.
for $\{x,y\} \in E(G) - E(T)$ do
    merge all sets containing vertices in $L(x,y)$
end do
```

Since the vertices are labeled in topologically sorted order, we can show the following.

Lemma 4.1. At the beginning of an iteration of the loop in the basic algorithm, let $S(u)$ be the largest vertex in the set containing u. Then each iteration of the loop preserves the following property: for each vertex u, either $S(u) = S(p(u))$ or $p(u) = p(S(u))$.

Proof. Note that an equivalent statement of the property is: $p(S(u))$ is an ancestor of u in T and all of $L(u,p(S(u)))$ is in the same set. The property holds initially, so assume that it holds at the beginning of an iteration for cotree edge $\{x,y\}$. Let u be any vertex. Let $S'(u)$ denote the largest value in the set containing u at the *end* of the current iteration. Note that if $S(u) = S'(u)$ or if $S(u) = S(p(u))$ the property of the lemma is preserved. If $S(u) \neq S(w)$ for any $w \in L(x,y)$, then $S'(u) = S(u)$.

So, assume that $S(u) = S(w)$ for some $w \in L(x,y)$ and that $S(u) \neq S(p(u))$. Because the property holds for S and u, $p(u) = p(S(u))$. Let w' be the smallest ancestor of w such that $p(w')$ is not in the same set as w. Then since

$$p(w') = p(S(w')) = p(S(u)) = p(u),$$

$p(u)$ is an ancestor of w and all vertices of $L(w,p(u))$ are in the same set.

We consider two cases.

(i) $p(u) \in L(x,y)$: Then after the merge, u,w, and $p(u)$ are all in the same set, so $S'(u) = S'(p(u))$.

(ii) $p(u)$ is an ancestor of $lca(x,y)$: We know there exists $v \in L(x,y)$ with $S(v) = S'(u) > S(u)$. Let $t = p(S(v))$. Then t is an ancestor of v and all of $L(v,t)$ is in the same set at the beginning of the iteration. If $t = p(u)$, then $p(u) = p(S'(u))$ and the lemma is preserved. If $t < p(u)$, then $t \in L(w,p(u))$ and $S(t) = S(u)$. But this implies

$$S(t) \geq t > S(v) > S(u) = S(t),$$

a contradiction. If $t > p(u)$, then $p(u) \in L(v,t)$ and $S(p(u)) = S(v)$, i.e. $S'(p(u)) = S'(u)$ and the lemma is preserved. □

Rather than attempting to do the lca searches on T, we can do them on a modified tree T'. The exact details of how T' is defined are different for the array and sequential implementations of the algorithm. However, we can think of T' as being the same as T with all the vertices that belong to the same equivalence class contracted into one vertex. The novelty of our algorithm is that as it progresses, T is modified (to become T') so that all equivalence classes are subtrees of T', and yet merging along fundamental cycles wrt T' has the same effect as merging along fundamental cycles wrt T. Thus a single tree T' represents the information computed so far about equivalence classes as well as all information required to merge equivalence classes for subsequent fundamental cycles wrt T.

Equivalence classes themselves are rooted subtrees of T'. We accomplish the actual set merging by putting vertices of the same class into the same tree of a forest which we will refer to as F. Let $p'(v)$ denote the parent of v in T' and let $f(v)$ denote the parent of v in F. Define T' by the relation:

$$\begin{aligned} p'(v) &= p(v) \text{ if } v \text{ is a root in } F \text{ (in which case } f(v) = 0), \\ &= f(v) \text{ otherwise.} \end{aligned}$$

The structure of T' agrees as much as it can with the structure of T, and the ordering of T' is consistent with that on T:

(&) For every vertex v, $v < p'(v)$.

For vertices x,y, let $z = lca'(x,y)$ be the lowest common ancestor of x and y in T'. Let $L'(x,y)$ denote the list of all vertices which are ancestors in T' of x or y but proper descendants in T' of z After each iteration of the modified algorithm below, F is a directed in-forest.

Modified Algorithm

```
T' := T (each vertex is a separate tree in F)
for $\{x,y\} \in E(G) - E(T)$ do
    merge all trees in F containing vertices of $L'(x,y)$ into a single
        tree in such a way that property (&) is preserved
end do
```

During an iteration for the cotree edge $\{x,y\}$, note that the sets merged by the original algorithm are exactly the sets containing vertices in

$$\{S(u) | u \in L(x,y)\}.$$

Similarly, the modified algorithm merges exactly those sets containing vertices in

$$\{S(u) | u \in L'(x,y)\}.$$

The following lemma (which is a loop invariant for the modified algorithm) shows that the partition into equivalence classes induced by the modified algorithm agrees with that of the basic algorithm. (For a proof, see [11].)

Lemma 4.2. At the beginning of every iteration of the loop in the modified algorithm, $\{S(u) | u \in L(x,y)\} = \{S(u) | u \in L'(x,y)\}$ for every pair of vertices x,y. □

The process of merging along paths in T' can be implemented efficiently on a one-way one-dimensional array of processors. The objects to be merged are equivalence classes represented by trees in the forest F. Merging two classes will always be performed by making the root of one of the trees become a child of *some* vertex in the other tree. Therefore, once a vertex has a nonzero parent in F, that parent will never be changed.

Regard the list $L'(x,y)$ as being in topologically sorted order. The forest F will be modified by an edge (x,y) as follows. Let $z = lca'(x,y)$ and let l be the last node of the ordered list $L'(x,y)$. Each vertex v on $L'(x,y)$ which is the root of a tree in F (that is, $S(v) = v$) will become linked in F to its successor on $L'(x,y)$, except possibly for l. If l is not the root of a tree in F, the parent in F of l will not change. Otherwise, $p'(l) = z$ and two cases can occur.

Case 1. There exists a vertex k distinct from l in $L'(x,y)$ such that $p'(k) = z$ and k is not the root of a tree in F. In this case l will be linked in F to z.

Case 2. All other possibilities. In this case, l will not be linked in F to anything by the merge.

We now describe the biconnectivity algorithm for the array. Assume that a spanning tree T of the graph G has been found, as in [4], and that it has been rooted, as described in Section 3, to obtain T. At the cost of a small amount of extra processing, assume that all vertices of G are identified by their numbers in the topological ordering with respect to T. Each cell of the array represents a vertex; cell w represents vertex w. Initially each cell w stores its parent $p(w)$ in T and its parent $f(w)$ (initially 0) in F. Since $p'(w)$ is $p(w)$ if w is a root in F and $f(w)$ otherwise, parents in T' need not be stored explicitly. Edges of the graph G enter cell 1 of the array and, because of the one-way data flow through the array, may enter the array at successive time units. As each edge $\{x,y\}$ of G passes through the array, it is transformed to reflect the vertices along $L'(x,y)$. It updates the zero parents in F of those vertices in order to merge their equivalence classes. The edge must carry with it the information $(ucase, vcase)$ to determine if the last vertex in $L'(x,y)$ should be joined in F to its parent--to see if $L'(x,y)$ is in *case 1* or in *case 2*.

When the record $(u,v,ucase,vcase)$ enters cell w, $u = v = 0$ if $w \geq z$. If $w < z$ then u is the minimum vertex of $L'(x,z)$ which is larger than or equal to w (if there is none then $u = 0$). Furthermore $ucase = 2$ iff either $u = x$ or the child of u along the path from x to z in T' is a root in F. The same conditions hold for v wrt $L'(y,z)$.

The program executed by each cell w is described on the next page. Its correctness follows from the correctness of Lemma 4.2 and from detailed arguments in [11].

After all edges of G have passed through the array, both the tree T and the forest F are recorded in the array. The final state of the array may be used to output each tree edge tagged with its biconnected component, to output articulation points and bridges, or to tag all edges of the graph with their component number.

Array Algorithm for Biconnectivity

```
(u,v,ucase,vcase) := INPUT;
    if $w = u$ then
        if $p'(w) = v$ then
            /* $w = l$ and $v = z = p'(k)$. Further, $L'(x,y)$ is in case 1
                if and only if $vcase = 1$. */
            if $vcase = 1$ then
                $f(w) := v$; end if
            $(u,v) := (0,0)$
        else /* $p'(w) \neq v$ */
            $u := p'(w)$;
            if $f(w) = 0$ then
                $f(w) := \min\{u,v\}$;
                $ucase := 2$
            else
                $ucase := 1$ end if
        end if
    else if $w = v$ then
        /* INSERT SAME CODE AS FOR $w = u$, EXCEPT INTER-
          CHANGE OCCURRENCES OF $u,ucase$ WITH $v,vcase$ */
    end if
end
```

It is worth noting that the modified algorithm also has an efficient sequential implementation. We can use standard disjoint set merging techniques to maintain the value of $S(v)$ for each vertex v. Then F can be defined by the following relation:

$$f(v) = S(v) \text{ if } S(v) \neq v$$
$$= 0 \text{ otherwise}$$

It is easy to see that this definition obeys property (&). Lemma 4.2 shows that we can do the lca search in T' rather than in T. If the operations required to maintain disjoint sets are ignored and the lca searches in T' are done so that the number of vertices accessed during the search is no more than $2\cdot|L'(x,y)|$, then our implementation runs in linear time.

Note that prior to an iteration of the modified algorithm, the number of distinct sets (distinct values of $S(v)$) among the vertices of $L'(x,y)$ is at least $(k-1)/2$ where $k = |L'(x,y)|$. Each iteration involves at most $3k$ accesses of forest pointers ($2k$ for the lca search and at most k to merge all the sets). Bookkeeping operations can easily be bounded by $O(k)$. Another way of looking at it is that the cost of every iteration is at most $c_1 + c_2 j$, where c_1 and c_2 are constants and j is the decrease in the total number of sets that occurs during the iteration. Since the sum of all the j's is at most $n-1$ (we start with n sets and end up with at least 1), we conclude that the algorithm does $O(m+n)$ operations if disjoint set maintenance is ignored.

If disjoint set operations are taken into account, the worst case running time of the algorithm is actually $O(m\alpha(m,n))$. However, several simple implementations of disjoint set operations have expected running time of $O(m)$ (see [5],[9]). If this algorithm is used in conjunction with an algorithm that also reads the edges of the graph sequentially and uses disjoint set operations to construct T, then it has the practical advantage that edges are always accessed sequentially and only $O(n)$ (versus $\Omega(m)$ for traditional graph algorithms) random access storage is required.

If T is given, the disjoint set operations can be implemented to run in time $O(m+n)$ [6]. This is of theoretical interest only, to show that our algorithm can be implemented to run in linear time like other biconnectivity algorithms.

References

[1] S. Ashtaputre, Data structures for solving graph problems on a one-dimensional array, Master's Thesis, Computer Studies, NCSU, 1983.

[2] M.J. Atallah and S.E. Hambrusch, Solving tree problems on a mesh-connected processor array, Proceedings, 26th Annual Symposium on Foundations of Computer Science, pp. 222-231, October 1985.

[3] M.J. Atallah and S.R. Kosaraju, Graph problems on a mesh-connected processor array, JACM, Vol.31, No.3, pp. 649-667, July 1984.

[4] S. Ashtaputre and C. Savage, Systolic arrays with embedded tree structures for connectivity problems, IEEE Transactions on Computers, Vol. C-34, No. 5, May 1985.

[5] B. Bollobas and I. Simon, On the expected behavior of disjoint set union algorithms, Proc. 17th Ann. ACM Symp. on Theory of Computing, pp. 224-231.

[6] H.N. Gabow and R.E. Tarjan, A linear time algorithm for a special case of disjoint set union, JCSS, Vol.30, pp. 209-221.

[7] S.R. Kosaraju and M.J. Atallah, Optimal simulations between mesh-connected arrays of processors, Proceedings, Eighteenth Ann. ACM Symposium on Theory of Computing, 264-272, May 1986.

[8] H.T. Kung and M.S. Lam, Fault-tolerance and two-level pipelining in VLSI systolic arrays, Proc. 1984 Conference on Advanced Research in VLSI, M.I.T. (ed. Paul Penfield, Jr.), 74-83.

[9] D.E. Knuth and A. Schönhage, The expected linearity of a simple equivalence algorithm, Theoretical Computer Science, Vol.6, pp. 281-315.

[10] C. Savage and M. Stallmann, Decomposability and fault tolerance in one-dimensional array algorithms, in preparation.

[11] C. Savage, M. Stallmann, J. Perry, Solving combinatorial problems on arrays with one-way dataflow, NCSU-CSE Technical Report 86-12.

[12] M. Tchuente and L. Melkemi, Reseaux systoliques pour le calcul des composantes connexes et la triangularisation des matrices bandes, R.R. No. 366, Laboratoire d'Informatique et de Mathematiques Appliquees de Grenoble, March, 1983.

[13] R.E. Tarjan and U. Vishkin, An efficient parallel biconnectivity algorithm, SIAM J. Computing, Vol.14, No.4, pp. 862-874, November 1985.

A PARTITIONING PROBLEM ARISING IN THE DESIGN OF SMALL UNIVERSAL EMULATORS

Richard Newman-Wolfe
Computer Science Department
The University of Rochester
Rochester, NY 14627
now at
Computer and Information Sciences Department
The University of Florida
Gainesville, FL 32611

ABSTRACT

A graph partitioning problem related to the undirected graph dominator problem is presented. This problem arises in the construction of communication networks when the architecture is constrained. Here, we seek small communication networks, or connectors. It is shown that for arbitrary graphs, requiring the connector to be a connected component of the graph can force it to be at most one node larger than the smallest possible connector. Lower bounds on the number of nodes required for connectors are given for general graphs and for the n-cube. A construction for the n-cube yields an upper bound less than twice as large as the lower bound developed.

INTRODUCTION

This paper considers a partitioning problem arising in the design of small switching networks. The switching network must provide communication paths between each pair of processing elements (PE's) for a universal emulator [Preparata '84]. If the objectives are to conform to certain connectivity constraints and to keep the size of the communication medium small, the design problem may be approached as partitioning the nodes of a fixed architecture into PE's and switching elements (SE's).

The problem of choosing SE's and PE's may be formalized as the following general graph partitioning problem:

Definition: Let $G = <V,E>$ be a graph and $P=\{A,B\}$ be a partition of V. If, for each pair of nodes, v, w, in A, there is a path between v and w not including any other nodes of A, then P is said to *have the outside path property, A* is said to have *outside paths* and B is said to *provide the outside path property*. B is also called a *connector*. If there is a non-empty path in B between v and w, then P has the *strict outside path property, A* has *strict outside paths* and B *provides the strict outside path property*. B is then called a *strict connector*. If B is connected, then it is a *connected connector*.

The general problem is related to the *dominating set* problem for undirected graphs. The dominating set problem for graph $G=<V,E>$ and an integer K is to find whether there is a subset $V' \in V$ of at most K nodes such that every omitted vertex $v \in V - V'$ is joined to at least one vertex in V' by an edge in E. This problem has been shown to be NP-complete [Garey & Johnson '79]. Garey and Johnson have shown that the variant in which V' is required to be connected is also NP-complete. It is clear that a connected dominating set will provide the strict outside path property, and that a part providing the strict outside path property will be a dominating set. If the dominating set is unconnected, the problem is the same as finding the "spines" of the upper bound construction given later.

The lower bounds results presented here are interesting in light of the fact that they allow us to answer easily some of the questions about dominating sets (in the negative). In particular, they place lower bounds on the size of dominating sets.

The first section demonstrates that connected, strict connectors are no more than one node larger than the smallest connector. The next section uses this fact to establish lower bounds for arbitrary graphs and n-cubes. The proofs of these appear in [Newman-Wolfe '86]. Finally, a construction technique that is asymptotically optimal for the n-cube is presented; it has B less than twice as large as the lower bound.

CONNECTEDNESS IS COST EFFECTIVE

For *any* connected graph, the smallest connected connector is at most one node larger than the smallest connector. Sometimes it *does* require one node more to have the connector connected (for example, a ring of six nodes has this property).

Theorem 1: *For each connected graph* G=<V,E>, *there is a partition* P={A,B} *such that* B *is a connected, strict connector and is no more than one node larger in size than the smallest unconnected, strict connector for* G.

Proof Sketch: Start with a minimal, strict connector B. By adding at most one node of A to B, a single connected component of B is created that includes all of the nodes of A in its neighborhood. Each other connected component of B can then be merged with this one universal component by exchanging nodes between A and B, without a further net increase in the size of B. The proof may be broken into the following three lemmas.

Lemma 2: *If* A *has strict outside paths, then for each node* v *in* A, (a) A'=A–{v} *has strict outside paths (of course) and* (b) *every node in A' has an edge to some connected component of* B *adjacent to* v.

Lemma 3: *If* B *provides strict outside paths for* A, *then, for any node* v *in* A, B'=B∪{v} *has a single connected component that provides strict outside paths for* A–{v}.

Lemma 4: *If there is a partition* P={A,B} *such that* A *has strict outside paths and there is some single connected component* B_1 *of* B *to which all nodes of* A *are adjacent, then there is a partition* P'={A',B'} *such that* A' *has strict outside paths,* B' *is connected* *and* B' *is no larger than* B.

Thus in any graph, including binary cubes, if we seek a strict connector we may as well look for a connected, strict connector, knowing that it will cost us at most one node over the optimal. In fact, this is true even if we are looking for a non-strict connector.

Corollary 5: *For each connected graph* $G=<V,E>$, *there is a partition* $P=\{A,B\}$ *such that* B *is a connected, strict connector and is no more than one node larger in size than the smallest unconnected, non-strict connector for* G.

The proof of this proceeds in a manner similar to the preceding one, but the first two lemmas must be changed slightly.

Lemma 6: *If* A *has outside paths, then, for each node* v *in* A, (a) $A'=A-\{v\}$ *has outside paths (of course) and* (b) *every node in* A' *has an edge to* v *or to some connected component of* B *adjacent to* v.

Lemma 7: *If* B *provides outside paths for* A, *then, for any node* v *in* A, $B'=B\cup\{v\}$ *has a single connected component that provides strict outside paths for* $A'=A-\{v\}$.

LOWER BOUNDS

With the knowledge that having *B* connected cannot cost much, lower bounds for both arbitrary graphs and for binary hypercubes may be easily obtained. This section demonstrates that any connector for the n-cube must be of at least $\Omega(2^n/n)$. In general, $\Omega(n/\text{degree}(G))$ nodes will be required for a connector of any N-node graph G, where *degree(G)* is the maximum degree of the graph G.

Theorem 8: *For graph* $G=<V,E>$, *any connected, strict connector must have at least* $\lceil (N-2)/(d-1) \rceil$ *nodes, where* d *is the maximum degree of* G *and there are* N *nodes in* V.

Proof: Each node of connected, strict connector *B* can have at most *d* neighbors, hence if *B* is of size *k*, it has at most *dk* (directed) edges. Of these, at least $2(k-1)$ are required to connect the nodes of *B* (counting each undirected edge as two directed edges). The total size of *V* must be no larger than *k* plus the remaining edges, or $N \leq k+dk-2(k-1)$. Hence $k \geq (N-2)/(d-1)$.

Corollary 9: *For graph* $G=<V,E>$, *any connector must have at least* $\lceil (N-2)/(d-1) \rceil -1$ *nodes, where* d *is the maximum degree of* G *and there are* N *nodes in* V.

The proof follows from Corollary 5 and the preceding theorem. □

Making use of the fact that any two nodes in the n-cube at a distance of two apart must share two nodes in their neighborhoods, the lower bound on the size of a connector for the n-cube can be improved.

Corollary 10: *For the n-cube, any connector must have at least* $\lceil (2^n-4)/(n-2) \rceil -1$ *nodes, for* $n>2$.

AN UPPER BOUND FOR THE N-CUBE

A partitioning method yields a connected connector for the n-cube of size 0(N/log N), which is asymptotically optimum. The fundamental idea in the construction is to divide the hypercube into components based on some of the bits of the addresses of the nodes, and then assign all of the nodes in some of the components to be the "spines" of the set *B*. The spines will provide access to *B* for the nodes of *A*. Their choice uses perfect one-bit error-correcting Hamming codes [Hamming '80]. To these we add some bridges so that the set *B* is connected. The use of "bridge nodes" reduces the need for outside paths to mere accessibility in a lower dimension.

Theorem 11: *The n-cube may be partitioned with the symmetric outside path property and with the size of the connector, B, as small as* $2^n/(b+1)+2^{b+1}/(b+1)-2$*, where b is one less than a power of two and also less than or equal to n. For n a power of 2, therefore, the size of B can be as small as* $(2/n)2^n-2$.

OPEN QUESTIONS

The upper bound constructions for the n-cube are not the best possible. Better connectors have been found for 5-cubes and 6-cubes, for instance. Finding a regular construction that improves the constructions presented here is a subject of further research. In particular, the bridge nodes are not used to provide access to nodes of A, but only to connect the spines. There is a tree-like bridge structure that reduces the diameter without increasing the size of the network.

Non-trivial constructions for other families of graphs, and lower bounds for them, are also desirable. The lower bound for general graphs is somewhat weak. The weakness lies with graphs with nodes of widely varying degree.

Some work has been done on analysis of the network properties of the given construction [Newman-Wolfe '85]. Improvements to the fault-tolerance and diameter can be made without sacrificing the general assymptotic result, but they do increase the size of the connector. What tradeoffs exist between the size of the connector and these properties?

Finally, can the results given be usefully applied to VLSI complexity problems?

REFERENCES

[1] Garey, Michael R. and David S. Johnson [1979], Computers and Intractability , W. H. Freeman and Co., San Francisco, CA, 1979.

[2] Hamming, Richard W. ['80], Coding and Information Theory , Prentice-Hall, Inc., Englewood Cliffs, NJ, 1980.

[3] Newman-Wolfe, Richard E. ['85], "*A Set Theoretic Problem: A Solution for the Hypercube and its Applications,*" TR 171, Computer Science Department, The University of Rochester.

[4] Newman-Wolfe, Richard E. ['86], "*Outside Path Partitions,*" in Proceedings of the 1984 Open House , TR 184, , R. E. Newman-Wolfe (ed.), Computer Science

Department, The University of Rochester, April 1986.

[5] Preparata, F.P. ['84], "*VLSI Algorithms and Architectures,*" Proceedings of the 11th Symposium on Mathematical Foundations of Computer Science , in Lecture Notes in Computer Science 176 , Goos, G. and J. Hartmanis, eds., Springer-Verlag, New York, 149-161, 1984.

This work was supported inpart by DARPA Grant DACA76-85-C-0001.

ASPECTS OF PARALLEL COMPUTATION IN GLOBAL OPTIMIZATION

PANOS M. PARDALOS

Computer Science Department
The Pennsylvania State University
University Park, PA 16802

ABSTRACT

Global optimization problems involve a large number of iterations, function evaluations and a large number of independent local searches. Concurrent function evaluations are well suited for multiprocessing. Similarly concurrent iterations used for local minimizers can be implemented using multiprocessors. We consider here the global minimization of nonconvex quadratic problems. Our approach is based on the use of linear programming. Specifically we solve multiple-cost-row linear programs and closely related nonlinear programs. In this approach, because the constraint set is unchanged and only the objective function changes the subproblems can be solved independently of each other. Therefore use of multiprocessor capabilities is an attractive possibility. We propose some alternatives and illustrate our approach by solving the Linear Complementarity Problem. Computational results are reported using the Cray 1S and Cray X-MP/48 supercomputers.

1. INTRODUCTION

Global optimization is concerned with the characterization and computation of global minima (or maxima) of nonlinear functions. We consider here the constrained global minimization problem that has the following form:

$$\text{global} \min_{x \in \Omega} f(x) \tag{1}$$

where the objective function $f(x)$ is continuous nonconvex, and the feasible domain Ω is a nonempty bounded polyhedron in R^n.

Such problems are widespread in mathematical modeling of real world systems for a very broad range of applications. Although standard nonlinear programming algorithms will usually obtain a local minimum to the problem, such a local minimum may not be global. In general several local minima may exist with corresponding function values that differ substantially.

From the computational complexity point of view (1) remains NP-hard even for such special cases as that of minimizing a quadratic concave function over the unit hypercube. For such problems heuristic approaches or approximate algorithms are inevitable as the size problem increases.

In the general class of (1) problems we are going to consider a certain subclass which seems to be more tractable. This class is very rich in practical applications and includes quadratic problems (concave and indefinite) and more generally separable objective functions. A large number of combinatorial optimization problems can be formulated as nonconvex quadratic programs. Certain aspects of physical VLSI chip design and database design problems are included in this class [7].

Global optimization problems are computationally very difficult. Although parallel processing by itself does not provide a solution to the problem, is rewarding when the problem is solved by heuristics or approximate algorithms in polynomial time.

Very little work has been done in constrained parallel global optimization, comparative to the other areas as for example parallel algorithms for linear algebra. Even for the case of linear programming very little work is done in that direction. With the exception of certain parallel linear algebra algorithms applied to linear programming (see for example [6]), very few parallel architectures have been suggested for optimization problems. In [2] a special architecture is discused for the parallel solution of linear programming using the Dantzig-Wolfe decomposition algorithm. Another method that takes advantage of low-level parallelism is discussed in [10]. For unconstrained optimization a number of parallel algorithms have been proposed (local methods). Some of these algorithms and a number of relevant references can be found in [SCHE84].

Global optimization problems involve a large number of iterations and function evaluations and often a large number of independent local searches. Concurrent function evaluations are well suited for multiprocessing. Similarly concurrent iterations used in search for local minimizers can be implemented using multiprocessors.

In recent years several multiprocessors have been developed in experimental environments and many of them are commercially available. Our approach for solving (1) is based on the use of linear programming. Specifically we solve multiple-cost-row linear programs and several closely related nonlinear programs (local search). Subproblems to be solved are independent of each other, hence their solution could logically proceed in parallel.

In the next section we demonstrate the use of of multiple-cost-row linear programming for global optimization of quadratic programs. Multiprocessor capabilities are very attractive, since for this approach the constraint data is unchanged and only the objective function chances.

Next, some methods are proposed for multiprocessing and we illustrate our approach by considering the Linear Complementarity Problem. Computational results are given using the Cray 1S and Cray X-MP/48 supercomputers.

2. USE OF MULTIPLE-COST-ROW LINEAR PROGRAMMING

Consider the quadratic global minimization problem,

$$\text{global} \min_{x \in \Omega} c^T x + \frac{1}{2} x^T Q x \tag{2}$$

where $\Omega = \{x : Ax \leq b \text{ , } x \geq 0\}$ is a bounded polytope in R^n, A_{mxn} matrix and Q_{nxn} is a symmetric matrix. If Q is negative definite then the global minimum occurs at some vertex of the polytope Ω. If Q has negative and positive eigenvalues then it is easy to see that the global optimum occurs at a boundary point of Ω, not necessarily an extreme point.

There are several ways to reduce problem (2) to an equivalent separable one. In our approach we preserve the eigenstructure of the original problem. This is an important fact since the error bounds depend on the eigenvalues of the matrix Q [9]. To carry out the reduction note that since Q is symmetric, $Q = UDU^T$, $U = [u_1, \ldots, u_n]$, $D = diag(\lambda_1, \ldots, \lambda_n)$ where λ_i are the eigenvalues of Q with corresponding eigenvectors u_i and $U^T U = I$. Solve now the multiple-cost-row (MCR) linear program,

$$\max_{x \in \Omega} u_i^T x = \Delta_i \text{ , } \min_{x \in \Omega} u_i^T x = \delta_i \text{ , } i = 1, \ldots, n \tag{3}$$

Using the affine transformation

$$x \leftarrow U^T x - \Delta$$

we obtain an equivalent problem in separable form. For simplicity of notation the separable problem has the form:

$$\text{global} \min_{x \in \Omega} \phi(x) = \sum_{i=1}^{n} \phi_i(x_i)$$

where $\phi_i(x_i) = d_i x_i + \frac{1}{2} \lambda_i x_i^2$, $i = 1, \ldots, n$ and $\Omega = \{x : Ax \leq b \text{ , } 0 \leq x_i \leq \beta_i \text{ , } i = 1, \ldots, n\}$. Here $\beta_i = \Delta_i - \delta_i$, $i = 1, \ldots, n$.

Solving the above (MCR) linear program we obtain vertices $v_1, \ldots, v_k$, $k \leq 2n$. Let $\phi_a(v) = \min_{1 \leq i \leq k} \phi(v_i)$, be an approximation to the global minimum. Computational experience with a large number of test problems indicates that by applying a local search with starting point v, we often obtain the optimum (or a good approximation), and in many cases v is actually itself the global optimum. Most of the computational effort is used to verify that this is the global solution, by improving lower and upper bounds in the separable program [8].

The next theorem will provide some insight to the above computational observation. By the mean value theorem there exist an $m > 0$ such that

$$|\phi(x) - \phi(y)| \leq m \; || x - y || , \; x, y \in \Omega$$

The constant $m = \max || \nabla \phi(x) ||$ for all $x \in \Omega$, and can be found easily since the function is separable. Let $v_1, \ldots, v_k$ be feasible points (usually local minima) and define ϕ_a as before. Then we have the following result:

Theorem 1: Consider the spheres $S_{v_i}(r_i)$ where $r_i = (\phi(v_i) - \phi_a)/m$, $i = 1, \ldots, k$, and suppose that $\bigcup_{i=1}^{k} S_{v_i} \supseteq \Omega$. Then ϕ_a is the global optimum.

Proof: Let $x \in \Omega$, then $x \in S_{v_{i0}}$ for some $i0 \in \{1, \ldots, k\}$ and therefore

$$|\phi(x) - \phi(v_{i0})| \leq m \; || x - v_{i0} || \leq m r_{i0} = \phi(v_{i0}) - \phi_a$$

Then $\phi(x) - \phi(v_{i0}) \geq -\phi(v_{i0}) + \phi_a$ and so $\phi(x) \geq \phi_a$ for all $x \in \Omega$.

For the case of concave objective function, the above result is true if $\cup S_{v_i}$ contains only the vertices of Ω. Suppose further that $\cup S_{v_i}$ does not contain Ω. Let $r_i \leftarrow r_i + \epsilon/m$. Call the new spheres $S_{v_i}^{\epsilon}$. Then the following is true:

Corollary: Let v_i be feasible points and suppose that $\cup S_{v_i}^{\epsilon} \supseteq \Omega$. Then ϕ_a is an ϵ-approximate solution in the sense that $\phi_a - \phi^* \leq \epsilon$ where ϕ^* is the global minimum.

Proof: Similar to the proof of the above theorem.

The above results although of theoretical interest, cannot be applied directly using the polyhedron Ω. Instead we can use a rectangle $R \supseteq \Omega$ to obtain useful information. Heuristics and methods for covering a rectangle with spheres are discussed in [3].

3. PARALLELISM FOR MULTIPLE-COST-ROW LINEAR PROGRAMS.

From the computational point of view we are interested to speed up the time required to solve the (MCR) program.

Cray computers offer two significant approaches in which parallelism inherent in an application can be exploited to improve performance. First, vector registers in the hardware allow certain computations to be carried out in parallel. Vectorization is a method by which a code is modified by the compiler to use this hardware feature. Second, multiple processors allow certain computations to be carried out in parallel on two or more processors. Multitasking is the method used to modify program code to use multiple processors.

In machines like the Cray 2 and Cray X-MP/48 multitasking is the technique that speeds up the execution time of a program. Multitasking is the structuring of a program into two or more parallel instruction streams (the tasks) that execute concurrently. Details on multitasking and related issues can be found in [1].

One obvious way to solve the (*MCR*) problem, is to have the same number of tasks as there are processors. A first objection to this approach is that does not reflect the inherent parallel nature of the problem since not all of the linear problems can be solved in parallel ($n >> 4$). Using the same number of tasks as there are processors, one processor is dedicated to each task until its completion time.

In machines with large number of processors we may consider each task to be a linear problem in (MCR). However very few machines are available with large number of processors. The theoretical speedup expected from multitasking can be predicted using Amdahl's Law.

4. LARGE SCALE GLOBAL OPTIMIZATION ON VECTOR COMPUTERS.

Let us consider now the quadratic global minimization problem formulated in Section 2. When Q is positive semidefinite then we have a convex problem and every local solution is global. However when Q is negative semindefinite or more generally indefinite then the problem is NP-hard.

Next we are going to illustrate our approach by considering a class of these problems called Linear Complementarity Problems (LCP):

Given M_{nxn} matrix, and $q \in R^n$

Find $x \geq 0$ such that $Mx + q \geq 0$, $x^T (Mx + q) = 0$

(or prove that such an x does not exist).

This problem has an equivalent formulation as a global optimization problem of the following form,

$$\text{global} \min_{x \in S} \phi(x) = \frac{1}{2} x^T \overline{M} x + q^T x$$

where $\overline{M}=M+M^T$ and $S=\{x: Mx+q \geq 0,\ x \geq 0\} \subseteq R^n$. Some of the important properties of LCP are summarized in the following theorems.

Theorem 2: If the LCP has a solution then it has a solution that occurs at a vertex of S.

Theorem 3: Suppose that there exists $a \in R^n$, $a \geq 0$ such that $a^T M < 0$. Then the feasible domain S is bounded (the converse is also true).

As a corollary it is easy to see that S is bounded if M is negative definite or if the sum of the elements of each column is negative. From now on we assume that S is bounded.

5. Outline of algorithm for LCP

A simplified form of the computational algorithm is a follows.

1. Choose n orthogonal directions u_i (for example the eigenvectors of $\overline{M}$) and solve the multiple cost row linear program

$$\min_{x \in S} \pm u_i^T x, \quad i=1, \ldots, n$$

This generates $k \leq 2n$ vertices v_i of S.

2. Evaluate $\phi(v_i)$, $i=1, \ldots, k$. If for some vertex v_j $\phi(v_j)=0$, stop; v_j solves the LCP.

3. Otherwise choose a subset of vertices with smallest function value. Use these vertices as starting points to get local solutions v_l usings MINOS [5]. If $\phi(v_l)=0$ stop, LCP is solved.

We make some remarks before we proceed. Almost all problems where solved at step 2 or 3 of the above procedure. In case a problem is not solved yet, then (by increasing the computational effort) we transform it to an equivalent mixed zero-one linear integer program. For details on this formulation see [8].

In several cases the LCP has no solution even if the feasible domain is nonempty. The problem of characterizing LCP with a nonempty feasible set that have no solution remains very difficult. A result that characterizes this situation is given by the next theorem:

Theorem 4: Suppose that $v_1, \ldots, v_k$ are vertices of S and m is a real constant satisfying $|\phi(x)-\phi(y)| \leq m \,||x-y||$. Let $\phi(v)=\min_{1 \leq i \leq k} \{\phi(v_i)\}$. Consider the spheres $S_{v_i}(r_i)$ where $r_i=(\phi(v_i)-\phi(v)+\alpha)/m$ and α is a positive number such that $\phi(v)-\alpha>0$.

If $\cup S_{v_i}(r_i) \supseteq S$ then the LCP has no solution.

Proof: Similar to the proof of Theorem 1.

6. COMPUTATIONAL RESULTS

For the computational results reported here we used the Cray 1S (with front-end the Cyber 845) and the Cray X-MP/48 (front-end IBM 4381). We generated test problems by generating the input data (M, q) in such a way that S is nonempty and such that a solution is known.

We used the best available software packages incorporated in the main program. To calculate eigenvectors we used EISPACK and MINOS 5.0 (Murtagh-Saunders) to solve linear and quadratic problems (MINOS obtains local solutions or stationary points when applied to the general quadratic problem).

With only 4 exceptions (out of a total 240) test problems, at least one solution was found to each problem. In some cases several distinct solutions were found.

Using the computational results we obtain a reasonably good fit to the data:

$$T_{AV} = (\frac{1.5}{10^4})n^4 \text{ secs}$$

Hence computationally the average performance of the algorithm is $O(n^4)$ with a reasonable constant.

First we consider computational results with sparse matrices. In the first table we have sparse matrices of bandwidth 3, and in the second bandwidth 5. In both cases we used the Cray 1S.

The table headings used are defined as follows:

$n =$ problem dimension, $T_{MCR} =$ CPU secs for the (MCR) linear program,
$T_{NLP} =$ CPU secs for nonlinear programs, $T_{AV} =$ CPU secs for each problem,
$T_{CR} =$ CPU secs per cost-row in the (MCR) linear program,
$N_{PIV} =$ Average number of pivot steps required for each cost row.

n	T_{MCR}	T_{TNP}	T_{AV}	T_{CR}	N_{PIV}
100	33.7	1.3	35.1	0.17	4
150	76.2	1.2	77.5	0.25	5
200	134.7	4.0	139.0	0.34	5
250	211.4	1.2	213.0	0.42	5
300	315.5	1.4	317.2	0.53	7
350	418.1	8.7	427.2	0.60	5
400	550.5	9.8	560.0	0.68	5
500	879.6	1.3	881.4	0.88	5

Table 1

n	T_{MCR}	T_{NLP}	T_{AV}	T_{CR}	N_{PIV}
100	46.7	0.69	47.6	0.23	18
150	7.6	0	7.8	0.33	12
200	211.7	1.04	213.0	0.53	27
250	271.1	9.46	280.9	0.43	16
300	406.7	1.98	409.1	0.68	18
400	905.9	3.48	910.0	1.13	33
500	1100	1.43	1102	1.10	16

Table 2

Next we consider computational results with dense matrices. In the first table problems were solved using the Cray 1S, and in the second table Cray X-MP/48.

n	T_{MCR}	T_{NLP}	T_{AV}	T_{CR}
10	0.46	0.052	0.55	0.023
20	2.93	0.33	3.39	0.073
30	9.96	1.16	11.11	0.16
40	28.1	3.36	32.18	0.35
50	83.2	10.4	94.3	0.83
60	155.5	15.1	172.0	1.30
70	293.6	34.9	329.8	2.10
80	643.	14.6	659.4	4.02
90	765.5	72.9	840.67	4.25

Table 3

n	T_{MCR}	T_{NLP}	T_{AV}	T_{CR}
10	0.33	0.033	0.39	0.016
20	2.05	0.155	2.31	0.05
30	7.15	1.19	8.57	0.12
40	20.3	1.50	22.15	0.25
50	43.9	4.07	48.55	0.44
60	85.9	5.86	92.57	0.72
70	207.6	23.6	232.3	1.48
80	425.9	48.6	476.1	2.66
90	673.4	46.3	721.8	3.74

Table 4

To compare the Cray 1S and Cray X-MP/48, we compare the last two tables. We see that the X-MP/48 is uniformly faster with a decrease in CPU time depending on the problem. For these preliminary runs no advantage was taken of the multiprocessor capabilities of the Cray X-MP/48. Each problem was automatically scheduled by the COS to one of the 4 processors, so that in effect CPU time is equivalent to solving on a single processor machine. On the average the CPU time solution was nevertheless reduced by a factor of approximately 0.7 when using the X-MP/48 compared to 1S. Substantial further reductions should be possible by taking full advantage of the multiprocessor capabilities and techniques proposed in earlier sections. We are in the process of investigating these capabilities.

Acknowledgement: The computational results reported here were obtained at the University of Minnesota and supported in part by NSF grant DCR8405489 to Prof. J.B. Rosen.

REFERENCES

[1] Cray Research Ins., A multitasking user guide (SN-0222). Mendota Heights (1985), MN.

[2] Dutta A., Siegel H.J. and Whinston A.B., On the application of parallel architectures to a class of operation research problems. R.A.I.R.O Res. Operationnelle/Oper. Res. Vol. 17, No. 4 (1983), pp. 317-341.

[3] Evtushenko Y.G., Numerical optimization techniques. Optimization Software Inc. (1985), N.Y.

[4] Kadmoum B., Speeding up the primal simplex algorithm on parallel computer. SIGMAP Bulletin, Vo. 31 (1982), pp. 19-23.

[5] Murtagh B.A. and M.A. Saunders, MINOS 5.0 User's Guide. Techn. Report SOL 83-20 (1983), Dept. of Oper. Res. Stanford Univ.

[6] Pan V. and J. Reif, Fast and efficient algorithms for linear programming and for the linear least squares problem. Technical Report TR-11-85, Aiken Comp. Lab., Harvard Univ. (1985).

[7] Pardalos P.M. and J.B Rosen, Methods for global concave minimization: A bibliographic survey. SIAM Review Vol 28, No 3 (1986), 367-379.

[8] Pardalos P.M. and J.B. Rosen, Global optimization approach to the linear complementarity problem. Technical Report UMSI 85/10, Supercomputer Institute, Univ. of Minnesota, (1985).

[9] Rosen J.B. and P.M. Pardalos, Global minimization of large-scale constrained concave quadratic problems by separable programming. Mathem. Progr. 34 (1986), 163-174.

[10] Schendel U., Introduction to numerical methods for parallel computers. Ellis Horwood, Chichester Publishers, (1984).

A DECOMPOSITION METHOD FOR OPTIMAL LINK SCHEDULING

RICHARD G. OGIER
SRI International
Menlo Park, CA

Consider a multihop radio network in which each node has a single transceiver which cannot transmit and receive simultaneously. The topology of the network is represented by an undirected graph $G = (V, E)$, where V is the set of nodes and E is the set of links. We say that a link $[i, j] \in E$ is *active* if one of the nodes i and j is successfully transmitting to the other. We assume that a set of links can be simultaneously active if and only if no two links in the set are incident to the same node, i.e., if and only if the set forms a *matching*. This may be the case, for example, in a spread-spectrum radio network using CDMA.

A *link schedule* is a variable-slot TDMA scheme in which the set of links active in each time slot is a matching. Thus a link schedule is specified by a finite sequence of matchings along with the corresponding slot lengths. The *length* of a schedule is defined to be the sum of its slot lengths. A link schedule is said to satisfy a *link demand* $f = (f_e : e \in E)$ if each link e is active for a total of f_e time units. This paper considers the problem of finding a minimum length schedule which satisfies a given link demand. Hajek and Sasaki [1] provided an $O(|V|^5|E|)$ time algorithm for this problem. In this paper, a new decomposition method is presented and used to improve the above time bound to $O(|V|^4|E|)$. For sparse graphs, the bound is improved from $O(|V|^5 \log |V|)$ to $O(|V|^4 \log |V|)$.

The reader is referred to [1] for a complete description of the algorithm of Hajek and Sasaki. We will discuss only its pertinent aspects. Let $P \subset \mathbb{R}^E$ denote the *matching polytope* for the graph G, i.e., the convex hull of the set of all indicators I_M of matchings $M \subset E$. The algorithm of [1] is based on the simple observation that the problem of computing a minimum length schedule which satisfies a demand f is equivalent to the following problem: compute $\tau = \arg\min \{\sigma : \sigma^{-1} f \in P\}$ and express $\tau^{-1} f$ as a convex combination of indicators of matchings.

The algorithm of [1] makes repeated use of a method due to Padberg and Rao [2] for determining a hyperplane (if one exists) which separates P from a given vector $h \in \mathbb{R}^E$. In this method, a set S is found which maximizes

$$w(h, A) \overset{\Delta}{=} \sum_{e \in L(A)} h_e - r(A)$$

over all sets $A \subset V$ having odd cardinality, where $r(A) = [|A| - 1]/2$ and $L(A)$ denotes the set of all links in E with both endpoints in A. If $w(h, S) > 0$, then $h \notin P$ and the hyperplane $\{x : w(x, S) = 0\}$ separates P from h. The set S can be found in $O(|V|^4)$ time by finding a minimum odd cut-set in a certain graph. (For sparse graphs, the method of Padberg and Rao requires $O(|V|^3 \log |V|)$ time.)

The algorithm of [1] suffers from the fact that the above method may be applied a large number of times, i.e., a large number of minimum odd cut-sets may be computed. We next present a new decomposition method which greatly reduces this number.

The method works as follows. Given a link demand vector $f = (f_e : e \in E)$, let σ be a positive number such that $\sigma \geq d(f, i)$ for all nodes i, where $d(f, i)$ is the sum of the demands of all links incident to i. Now let S minimize $w(\sigma^{-1} f, A)$ over all sets A of odd cardinality. If $|S| < |V| - 1$ and $w(\sigma^{-1} f, S) > 0$, then we define two new networks: G_1 is obtained from G by shrinking S to a single node v_1 (just as blossoms are shrunk in [3, p. 230]), and G_2 is obtained from G by shrinking $V - S$ to a single node v_2. Thus, each link from S to $V - S$ in the original network corresponds to a link incident to v_1 (resp. v_2). The demand for each link in each of the new networks is the same as that for the corresponding link in the original network.

Theorem. The minimum schedule length for each of the new networks is no greater than that for the original network. Furthermore, since at most one link incident to v_1 (resp. v_2) can be active at any time, the minimum length schedules for the two new networks can be combined to form a minimum length schedule for the original network by reordering the matchings of the two schedules so that at each time, a link incident to v_1 is active if and only if the corresponding link incident to v_2 is active.

Encountered within the algorithm of [1] is a problem of the form: compute $\tau =$ arg min $\{\sigma : \sigma^{-1} g + I_M \in P\}$ and express $\tau^{-1} g + I_M$ as a convex combination of indicators of matchings. This is actually a generalization of the original problem (since M may be the empty set), and the above decomposition method is easily extended to this problem.

The above method is used roughly as follows to compute a minimum length schedule. Proceed as in the algorithm of [1], except that whenever a minimum odd cut-set is computed to maximize $w(h, A)$ for some h, and the resulting set S satisfies $|S| < |V| - 1$ and $w(h, S) > 0$, apply the above decomposition method, i.e., solve the scheduling problems for the two smaller networks and then combine them. The improved time bound is due to the fact that at most $O(|E|)$ minimum odd cut-sets are computed, whereas in the algorithm of [1], as many as $O(|V|\,|E|)$ may be computed.

The above results have been extended to the case where each node may have several transceivers, so that several links incident to the same node can be simultaneously active. Details for both the single and multiple transceiver cases will appear in a forthcoming paper.

References

[1] B. Hajek and G. Sasaki, "Link Scheduling in Polynomial Time," submitted for publication.

[2] M. W. Padberg and M. R. Rao, "Odd Minimum Cut-Sets and b-Matchings," *Mathematics of Operations Research,* vol. 7, no. 1, pp. 67-80, 1982.

[3] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity,* Prentice-Hall, 1982.

A PERIODIC SCHEDULING PROBLEM IN FLOW-CONTROL FOR DATA COMMUNICATION NETWORKS

UTPAL MUKHERJI

AT&T Bell Laboratories, Murray Hill, NJ 07974

ABSTRACT

The scheduling of user-session transmission-priority time-slots, in time-periodic frames on links in a network, is considered from an algorithmic view-point. Priority-slot schedules with low schedule-delays can be used in a flow-control scheme to lower limits on packet intra-network delay. An NP-completeness result is proven, showing, for general networks, that the scheduling of priority-slots to obtain minimum sum of schedule-delays is algorithmically hard. Minimum-delay scheduling algorithms for special network classes, and a scheduling heuristic for general networks, are presented.

1 INTRODUCTION

1.1 Introduction

A schedule-based approach for flow-control in data communication networks has been described in Mukherji [8]. Some physical and architectural assumptions are inherent in the approach. First, each link can reliably transmit a known amount of information in a link-frame. Frames occur periodically on each link at intervals of one frame-time, where the frame-time is the same on all links. Second, each session has a path set up in the network at session set-up time, along which packets for the session can be routed. Packets for each session are transmitted in the same order on all links in the session's path. Third, information can be communicated to a session's source from its destination for flow-control purposes.

It is shown in Mukherji [8] that the schedule-based scheme provides each session in the network with a packet throughput value up to which throughput is guaranteed, as well as an achievable upper bound on the intra-network delay that a packet incurs in the network after starting transmission on the first link in its path. The throughput guarantee is obtained by assigning transmission priority to the session, on each link in its path, in certain time-slots in the link-frames. The upper bound on packet intra-network delay is obtained by, in addition, limiting the total number of packets for the session that can be in transit in the network at any given time, using an extended version of end-to-end windowing. The delay bound is shown to be a sum of, first, the product of the session's window-size and the frame-time, and, second, the session's schedule-delay. The schedule-delay is a function of the positions of the session's priority-slots in the link-frames.

The scheduling of priority-slots, so as to result in low values of schedule-delays, is considered from an algorithmic point of view in this paper. This problem is shown to be hard for general networks, as is the case for general instances of many other scheduling problems, by proving an NP-completeness result. (Garey and Johnson [3] provide a guide to the theory of NP-completeness.) However, simple and efficient algorithms that result in minimal schedule-delays are presented for special classes of networks. A scheduling heuristic, that can be applied to general networks, is also presented.

This work falls in the area of deterministic, time-periodic scheduling. Some work has been reported in this area (Orlin [9]), for example in the context of task scheduling in pipelined computers (Kogge [7], pp. 71-112) and of production scheduling in manufacturing flow-shops (Hitz [6], Graves et al. [4]).

1.2 Background

The schedule-based scheme is discussed briefly in this section.

The following network model is assumed.

a) All packets, for all sessions, are the same number of bits in length. Each link is time-slotted, with slot duration equal to the packet transmission time. Packet transmissions begin at the start-times of slots. Links are error-free, and can reliably communicate packets, one per time-slot. All links are equal in bit-speed, so that the slot duration on each link is the same. This duration is one time-unit.

b) All packets belonging to a session are routed along the same path in the network, where the path is chosen at session set-up time. The packets belonging to the session are transmitted in the same order on all links in the session's path. Packet propagation delays at all links, and packet switching delays at all nodes, are zero.

c) Window-tokens are returned to a session's source node from its destination node, for all sessions.

The frame-time for the network is chosen equal to T time-units, and a link-frame of length T slots is defined on each link. Each session is guaranteed a throughput of 1 packet per T time-units, as follows. At set-up time, a session's path is chosen subject to the restriction that the number of sessions sharing any link is at most T, and the session is assigned priority of transmission in one slot in the frame on each link in its path. Link-frames are repeated on their respective links at intervals of one frame-time. Thus, each session is guaranteed packet transmission priority, on each link in its path, at intervals of T time-units, making feasible a session throughput of 1 packet per T time-units.

An extended version of end-to-end windowing maintains the throughput guarantee for each session, while upper-bounding the intra-network packet buffering for the session, as described in Mukherji [8].

The intra-network delay, for a packet belonging to a session, is the time from the start of packet transmission on the first link in the session's path, to the finish of packet reception on the last link in the path. The packet intra-network delay is equal to 1 if the number H of links in the path is 1. Otherwise, as shown in Mukherji [8], the delay is upper-bounded by the sum of, first, the product wT of the session's window-size w and the frame-time T, and, second, the session's schedule-delay.

The schedule-sequence, schedule-delay, and schedule-wait, for a session, are defined as follows. Assume, for purposes of definition, that, first, the session can transmit only in priority-slots on each link in its path, second, the session has only one packet to carry through the network, and, third, the packet is transmitted on the first link in the priority-slot that starts at time σ_1, $0 \leq \sigma_1 < T$. Let σ_h, $2 \leq h \leq H$, denote the start-time of the priority-slot, on the h-th link in the path, in which the packet is transmitted on that link. Since priority-slots recur on each link at intervals of the frame-time T, the packet waits less than T time-units at each intermediate node in the session's path, i.e., $\sigma_h + 1 \leq \sigma_{h+1} < \sigma_h + 1 + T$, $1 \leq h \leq H - 1$. The schedule-sequence for the session is the sequence $\sigma_1, \sigma_2, \ldots, \sigma_H$. The schedule-delay for the session is $\sigma_H - \sigma_1 + 1$, the intra-network delay for the packet. The schedule-wait for the session is $\sigma_H - \sigma_1 + 1 - H$, the difference between the schedule-delay and the H time-units used for transmission of the packet. Since there are $H - 1$ intermediate nodes in the session's path, the schedule-wait is less than $(H - 1)T$. Hence, the schedule-delay is less than $(H - 1)T + H$, and the delay-bound, $(\sigma_H - \sigma_1 + 1) + wT$, is less than $(w + H - 1)T + H$.

1.3 Outline

A schedule that has non-integer valued slot start-times can be efficiently transformed into a schedule with only integer-valued start-times, without increasing the sum of schedule-delays; the procedure is described in Mukherji ([8], Appendix B). Accordingly, attention is restricted to the construction of schedules that have integer-valued slot start-times.

First, in Section 2, for networks that have frame-time equal to 3 time-units, the problem of deciding whether schedules that have all schedule-waits equal to zero exist, is shown to be NP-complete. In the theory of computational complexity, it is conjectured that there do not exist any polynomial-time algorithms for solving all instances of a problem that is NP-complete (Garey and Johnson [3]). Hence, some special classes of networks are considered in Section 3, for which linear-time algorithms are presented for computing schedules that have the minimum sum of schedule-waits. Finally, also in Section 3, a scheduling heuristic that can be applied to general networks is presented together with upper bounds on the resulting sum of schedule-waits.

This paper concludes, in Section 4, with additional comments on the periodic scheduling problem.

2 AN NP-COMPLETENESS RESULT

Let S denote the set of sessions, and L the set of links, in a network that has frame-time equal to T time-units. The paths for the sessions are known, and the number of sessions sharing any link is at most T. Let H^s, $1 \leq s \leq |S|$, denote the number of links in the path for the s-th session. The schedule for the network is the set of schedule-sequences for the sessions in S. More precisely, an integer schedule σ is a mapping σ_h^s, $1 \leq s \leq |S|$, $1 \leq h \leq H^s$, into the set of integers, that satisfies the following conditions 2.1.

$$0 \leq \sigma_1^s \leq T-1, \; 1 \leq s \leq |S|. \tag{2.1a}$$

$$1 \leq \sigma_h^s - \sigma_{h-1}^s \leq T, \; 1 \leq s \leq |S|, \; 2 \leq h \leq H^s. \tag{2.1b}$$

If the l-th link is the h-th sequential link in the s-th session's path and also the h'-th sequential link in the s'-th session's path, $1 \leq l \leq |L|$, $s \neq s'$, $1 \leq s, s' \leq |S|$, $1 \leq h \leq H^s$, $1 \leq h' \leq H^{s'}$, then

$$\sigma_h^s \neq \sigma_{h'}^{s'} \; (modulo \; T). \tag{2.1c}$$

The sum of schedule-waits for a schedule σ is

$$W_\sigma = \sum_{s=1}^{|S|} (\sigma_{H^s}^s - \sigma_1^s + 1 - H^s). \tag{2.2}$$

The following 'Network 3-Periodic Zero-Wait Scheduling' decision problem is shown here to be NP-complete : for a network instance I that has frame-time $T = 3$ does there exist an integer schedule σ that has sum of schedule-waits $W_\sigma = 0$? Some corollaries of this NP-completeness result are presented in Mukherji ([8], Appendix C).

Theorem 2.1 :

The 'Network 3-Periodic Zero-Wait Scheduling' problem is NP-complete.

Proof :

The theorem follows from a) and b) below.

a) The 'Network 3-Periodic Zero-Wait Scheduling' problem is in the class of problems NP. This is shown as follows.

Assume that there exists a zero-wait integer schedule σ for the network instance I. Then, the validity of conditions 2.1 for schedule σ, and of the value W_σ given by eqn. 2.2, can be checked in polynomial-time. Therefore, the problem is in the class NP.

b) The NP-complete 'Graph 3-Colourability' problem is reducible to the 'Network 3-Periodic Zero-Wait Scheduling' problem, in polynomial-time. This is shown below.

Let V denote the set of vertices, and E the set of edges, of a graph. A 3-colouring f for the graph is a mapping f_v, $1 \leq v \leq |V|$, into the set $\{0, 1, 2\}$, of 'vertex colours', that satisfies the following condition. If the e-th edge is $[u, v]$, i.e., incident on the u-th and v-th vertices, $1 \leq e \leq |E|$, $u \neq v$, $1 \leq u, v \leq |V|$, then

$$f_u \neq f_v. \tag{2.3}$$

Consider the 'Graph 3-Colourability' problem : for a graph instance $G = (V, E)$, does there exist a 3-colouring f ?

The procedure described next is used to construct a network instance I corresponding to the graph instance G, with the following properties. The network has frame-time $T = 3$. There is a one-to-one correspondence of sessions in the network to vertices in the graph. Each link in the network lies in one or two of the paths for the sessions. All links are numbered 0, 1, or 2. As shown in Fig. 2.1a, the numbers of the sequential links in each session's path constitute a modulo-3 count, with the first link numbered 0. The construction procedure is as follows.

i) For $1 \leq v \leq |V|$, repeat the following. Construct a separate first link, numbered 0, for the v-th session's path.

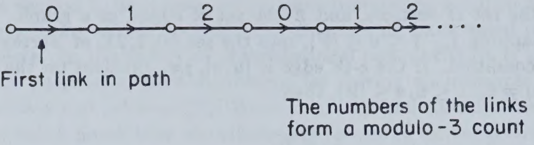
ii) For $1 \leq e \leq |E|$, repeat the following. If, in graph G, the e-th edge is incident on the u-th and v-th vertices, then extend the u-th and v-th sessions' paths so as to meet in a common link, as shown in Fig. 2.1b. If the old last links in the u-th and v-th sessions' paths are numbered x and y respectively, then $w(x, y)$ and $z(x, y)$ links are padded on in the respective paths, so that the number resulting for the new common last link forms a modulo-3 count with the link-numbers in each path.

This is a linear-time construction procedure; the constructed network I has $|V|$ sessions and at most $|V| + 5|E|$ links. As shown next, a zero-wait integer schedule exists for network I if, and only if, a 3-colouring exists for graph G. Thus, the 'Graph 3-Colourability' problem is reduced to the 'Network 3-Periodic Zero-Wait Scheduling' problem, in polynomial-time.

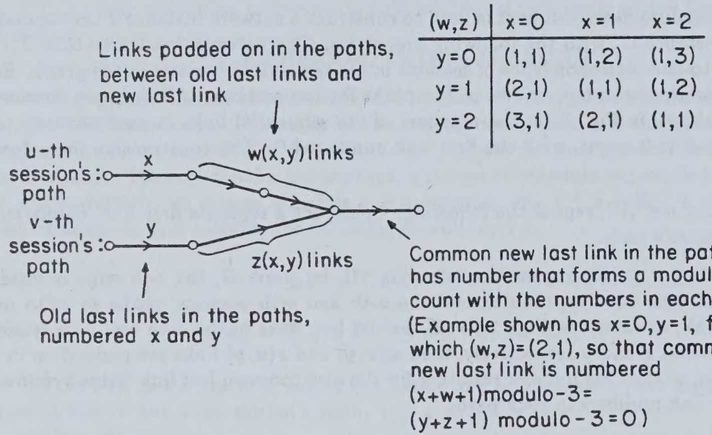
Suppose f is a 3-colouring for G. Then, as shown below, there exists a zero-wait integer schedule for I. Define σ by $\sigma_h^v = f_v + h - 1$, $1 \leq v \leq |V|$, $1 \leq h \leq H^v$. Then, the integers σ_h^v satisfy conditions 2.1a and 2.1b, and the value W_σ given by eqn. 2.2 is 0. Condition 2.1c is verified as follows. If a link is the h_u-th and h_v-th sequential link in the u-th and v-th sessions' paths respectively, then it is the common new last link resulting from the extension corresponding to the edge $[u, v]$ in G. Hence, $f_u \neq f_v(\textit{modulo } 3)$, and $h_u = h_v(\textit{modulo } 3)$. Therefore, $\sigma_{h_u}^u \neq \sigma_{h_v}^v(\textit{modulo } 3)$, and condition 2.1c is satisfied. Thus, σ is a zero-wait integer schedule for I.

Conversely, suppose that σ is a zero-wait integer schedule for I. Then, as shown below, there exists a 3-colouring for G. Define f by $f_v = \sigma_1^v$, $1 \leq v \leq |V|$. Then, each number f_v is 0, 1, or 2. Condition 2.3 is verified as follows. If $[u, v]$ is an edge in G, then there is a link in I that is the new last link resulting from the extension corresponding to edge $[u, v]$. Let this link be the h_u-th and h_v-th sequential link in the u-th and v-th sessions' paths respectively. Then, $\sigma_{h_u}^u \neq \sigma_{h_v}^v(\textit{modulo } 3)$, and $h_u = h_v(\textit{modulo } 3)$. Since $W_\sigma = 0$, $\sigma_{h_u}^u = \sigma_1^u + h_u - 1$ and $\sigma_{h_v}^v = \sigma_1^v + h_v - 1$. Therefore, $\sigma_1^u \neq \sigma_1^v(\textit{modulo } 3)$, and condition 2.3 is satisfied. Thus, f is a 3-colouring for G.

This completes the proof of Theorem 2.1.


0
1
2
0
1
2
. . . .
First link in path
The numbers of the links form a modulo-3 count

(a) The numbers of the links in a session's path



(w,z)	x=0	x=1	x=2
y=0	(1,1)	(1,2)	(1,3)
y=1	(2,1)	(1,1)	(1,2)
y=2	(3,1)	(2,1)	(1,1)

Links padded on in the paths, between old last links and new last link
u-th session's path
x
w(x,y) links
v-th session's path
y
z(x,y) links
Old last links in the paths, numbered x and y
Common new last link in the paths, has number that forms a modulo-3 count with the numbers in each path (Example shown has x=0, y=1, for which (w,z)=(2,1), so that common new last link is numbered (x+w+1) modulo-3 = (y+z+1) modulo-3 = 0)

(b) The extension of the u-th and v-th sessions' paths corresponding to edge $[u,v]$

Figure 2.1 **The Structure of the Network I in the Proof of Theorem 2.1**

3 SOME SCHEDULING ALGORITHMS

The discussion of scheduling algorithms is facilitated by the following definition of the link-precedence graph for a network. There is a one-to-one correspondence between links in the network and vertices in the link-precedence graph. There is a directed arc $a = (i, j)$ from the i-th vertex to the j-th vertex in the link-precedence graph if, and only if, at least one session in the network has the corresponding links, in the order (i, j), as consecutive links in its path. The weight $w_a = w_{i,j}$, of arc $a = (i, j)$ of the link-precedence graph, is the number of sessions that have the corresponding links as consecutive links in their paths.

If the link-precedence graph is a tree, then the following algorithm A_{tree} constructs an integer schedule that has sum of schedule-waits equal to 0, in linear-time. A_{tree} constructs the schedule link by link, each time scheduling all priority-slots that are to be scheduled in the link-frame under consideration. Figure 3.1 shows a network for which the link-precedence graph is a tree. Figure 3.2 shows a zero-wait schedule constructed for this network using A_{tree}.

Algorithm A_{tree} :

Step 1) Select a root vertex for the link-precedence tree. Number all $|L|$ vertices in the tree, in non-decreasing order of distance in link-hops from the root. Renumber links in the network so that each link-precedence graph vertex and its corresponding link have the same number.

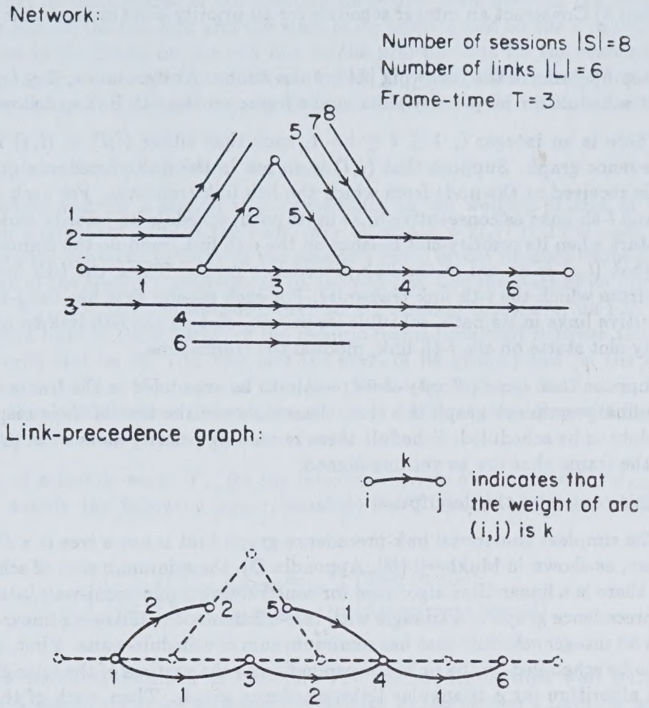

Network:
Number of sessions |S|=8
Number of links |L|=6
Frame-time T=3
5 7 8
1
2
2
5
1
3
4
6
3
4
6
Link-precedence graph:
k
i
j
indicates that the weight of arc (i,j) is k
2
2
5
1
1
3
4
6
1
2
1

Figure 3.1 The Link-Precedence Graph for a Network

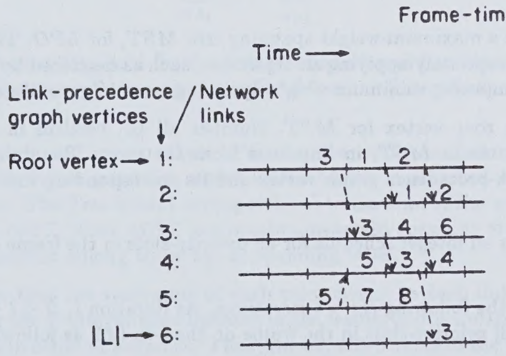

Frame-time T=3
Time ⟶
Link-precedence graph vertices / Network links
Root vertex → 1:
2:
3:
4:
5:
|L| → 6:
3
1
2
1
2
3
4
6
5
3
4
5
7
8
3

Figure 3.2 A Zero-Wait Schedule Constructed Using Algorithm A_{tree} for the Network of Figure 3.1

Step 2) Construct an integer schedule for all priority-slots in the frame on link number 1.

Step 3) Perform the following $|L| - 1$ iterations. At iteration l, $2 \leq l \leq |L|$, construct an integer schedule for all priority-slots in the frame on the l-th link as follows.

There is an integer i, $1 \leq i \leq l - 1$, such that either (i, l) or (l, i) is an arc in the link-precedence graph. Suppose that (i, l) is an arc in the link-precedence graph. Then, the i-th link is received at the node from which the l-th link transmits. For each session that has the i-th and l-th links as consecutive links in its path, schedule its priority-slot on the l-th link so as to start when its priority-slot finishes on the i-th link, modulo the frame-time. Suppose, instead, that (l, i) is an arc in the link-precedence graph. Then, the l-th link is received at the node from which the i-th link transmits. For each session that has the l-th and i-th links as consecutive links in its path, schedule its priority-slot on the l-th link so as to finish when its priority-slot starts on the i-th link, modulo the frame-time.

Suppose that some priority-slots remain to be scheduled in the frame on the l-th link. Since the link-precedence graph is a tree, these slots are the first of their respective sessions' priority-slots to be scheduled. Schedule these remaining priority-slots so as to start at integer times in the frame that are as yet unassigned.

This completes the description of A_{tree}.

The simplest non-trivial link-precedence graph that is not a tree is a directed triangle. In this case, as shown in Mukherji ([8], Appendix D), the minimum sum of schedule-waits is 0 or 1, and there is a linear-time algorithm for constructing a minimum-wait integer schedule. If the link-precedence graph is a triangle with tree-offshoots, the following linear-time algorithm computes an integer schedule that has minimum sum of schedule-waits. First, all priority-slots that are to be scheduled on the links corresponding to the vertices of the triangle are scheduled using the algorithm for a triangular link-precedence graph. Then, each of the tree-offshoots is scheduled using algorithm A_{tree}, with the appropriate triangle vertex already scheduled chosen as root vertex. The minimum sum of schedule-waits is either 0 or 1.

Algorithm A_{tree} can be extended so as to apply to general networks. The extended algorithm $A_{heuristic}$, described below, is applied to each connected link-precedence graph component LPG.

Algorithm $A_{heuristic}$:

Step 1) Compute a maximum-weight spanning tree MST, for LPG. This can be done in quadratic-time by appropriately applying an algorithm, such as described by Papadimitriou and Steiglitz [10], for computing minimum-weight spanning trees in general graphs.

Step 2) Select a root vertex for MST. Number all $|L|$ vertices in MST in non-decreasing order of distance,in MST, in link-hops from the root. Renumber links in the network so that each link-precedence graph vertex and its corresponding link have the same number.

Step 3) Construct an integer schedule for all priority-slots in the frame on link number 1.

Step 4) Perform the following $|L| - 1$ iterations. At iteration l, $2 \leq l \leq |L|$, compute an integer schedule for all priority-slots in the frame on the l-th link as follows.

Let A_l denote the set of arcs a in LPG, that are of the form $a = (i, l)$ (or $a = (l, i)$), where $1 \leq i \leq l - 1$. Let W'_a denote the following function of the schedule to be computed for the l-th link. If $a = (i, l)$, then W'_a is the sum, over all w_a sessions that have the i-th and l-th links as consecutive links in their paths, of the modulo-frame-time wait between the finish of the session's priority-slot on the i-th link and the start of its priority-slot on the l-th link. Otherwise, $a = (l, i)$, and W'_a is the sum, over all w_a sessions that have the l-th and i-th links as consecutive links in their paths, of the modulo-frame-time wait between the finish of the session's priority-slot on the l-th link and the start of its priority-slot on the i-th link. Assign integer start-times in the frame on the l-th link to the priority-slots for the sessions sharing the link, using an assignment algorithm that minimizes $\sum_{a\epsilon A_l} W'_a$. This can be done in cubic time by applying an optimal assignment algorithm such as described in Papadimitriou and Steiglitz [10].

This completes the description of $A_{heuristic}$.

The schedule-wait for a session that contributes to the weights of arcs in the link-precedence graph component LPG is the sum, over all arcs $a = (i,l)$ in LPG such that the i-th and l-th links are consecutive links in the session's path, of the modulo-frame-time wait between the finish of the session's priority-slot on the i-th link and the start of its priority-slot on the l-th link. W'_a, $a = (i,l)$, is the sum, over all w_a sessions that have the i-th and l-th links as consecutive links in their paths, of the modulo-frame-time wait between the finish of the session's priority-slot on the i-th link and the start of its priority-slot on the l-th link. The set of all arcs in LPG is the union of the $|L| - 1$ sets A_l, $2 \leq l \leq |L|$. Hence, the sum, over all sessions that contribute to the weights of arcs in LPG, of schedule-waits for sessions is $\sum_{l=2}^{|L|} \sum_{a\epsilon A_l} W'_a$.

The sum of schedule-waits W_σ, for the integer schedule σ computed by $A_{heuristic}$, is shown below to satisfy the following upper bounds. Let T denote the frame-time for the network. Then,

$$W_\sigma \leq \begin{cases} (T-1)\sum_{(i,j)\notin MST} w_{i,j} & \text{(Tree bound);} \\ \left(\frac{T-1}{2}\right)\sum_{(i,j)\epsilon LPG} w_{i,j} & \text{(Assignment bound).} \end{cases} \tag{3.1}$$

The Tree bound is obtained as follows. The modulo-frame-time wait between the priority-slots for a session that are on consecutive links in the session's path, in an integer schedule, is an integer between 0 and $T-1$. At iteration l in step 4 of $A_{heuristic}$, there is an integer i, $1 \leq i \leq l-1$, such that either (i,l) or (l,i) is an arc in MST. Suppose that priority-slots were scheduled at iteration l so that $W'_a = 0$ when $a = (i,l)$ (or $a = (l,i)$), as is done in algorithm A_{tree}. Then, defining A'_l to be $A_l - \{(i,l)\}$ (or $A_l - \{(l,i)\}$),

$$\sum_{a\epsilon A_l} W'_a \leq \sum_{a\epsilon A'_l} w_a(T-1). \tag{3.2}$$

This inequality must also hold for the actual scheduling at iteration l, for which the value of $\sum_{a\epsilon A_l} W'_a$ is minimum. Summing inequality 3.2 over all iterations l, $2 \leq l \leq |L|$, the Tree bound results.

The Tree bound is zero when LPG is a tree, and σ is then a schedule with zero sum of schedule-waits. More generally, a zero-wait schedule is obtained when the link-precedence graph is a forest. The Tree bound is proportional to the sum of the weights of all arcs in LPG that are not in MST. Since MST is a maximum-weight spanning tree for LPG, the value of this sum is minimum among those for all spanning trees.

Suppose that the start-time of each priority-slot in each link-frame is uniformly distributed among the T integer time-instants in the frame, and is independent of the start-times of priority-slots in other link-frames. Then, the expected value of the modulo-frame-time wait, between the priority-slots for a session that are on consecutive links in the session's path, is $(T-1)/2$. Hence, the expected value of the sum of schedule-waits that results from scheduling at random equals the Assignment bound.

In fact, the Assignment bound can be obtained using a similar argument, as shown below (Hajek [5]). An earlier proof is presented in Mukherji [8].

Consider the set of priority-slots that are scheduled at iteration l in step 4 of $A_{heuristic}$. Suppose that these priority-slots were scheduled so as to start at an arbitrarily chosen set of integer time-instants in the link-frame. Next, suppose that these priority-slots were rescheduled $T-1$ times, each time rotating the priority-slots by unit time to the right, cyclically in the frame. Then, averaging over all T schedulings, the start-time of each priority-slot would in effect be uniformly distributed among the T integer time-instants in the frame. Thus, averaged over all T schedulings,

$$\sum_{a \epsilon A_l} W'_a \leq \left(\frac{T-1}{2}\right) \sum_{a \epsilon A_l} w_a. \tag{3.3}$$

It follows that this inequality must hold for at least one of the T schedulings. Therefore, this inequality must also hold for the actual scheduling at iteration l, for which the value of $\sum_{a \epsilon A_l} W'_a$ is minimum. Summing inequality 3.3 over all iterations l, $2 \leq l \leq |L|$, the Assignment bound results.

Any integer schedule can be locally redefined without increasing the sum of schedule-waits by rescheduling a link, given the schedule on all its neighbouring links, using an optimal assignment algorithm. Suppose that links are considered for rescheduling in turn, keeping the schedule unchanged if rescheduling would leave the sum of schedule-waits unchanged, and rescheduling otherwise. Then, each rescheduling decreases the sum of schedule-waits by at least one. Suppose also that this iterative procedure is continued until each link has been considered for rescheduling at least once since the last rescheduling. Then, the sum of schedule-waits $W_{\sigma'}$ for the resulting schedule σ' also satisfies the Assignment bound. This result is obvious if the procedure above is followed starting with a schedule computed using algorithm $A_{heuristic}$. Otherwise, this result can be obtained by verifying the following statements. Inequality 3.3 holds when, in 3.3, the set A_l is replaced by the set A''_l of all arcs in LPG that are of the form (i,l) (or (l,i)). Suppose that schedule σ' is used for purposes of defining W'_a. Then, since an optimal assignment algorithm has been used at each iteration, inequality 3.3, with A_l replaced by A''_l, holds for $1 \leq l \leq |L|$. Further, $W_{\sigma'} = (\sum_{l=1}^{|L|} \sum_{a \epsilon A''_l} W'_a)/2$. Hence, from 3.3, $W_{\sigma'} \leq ((T-1)/2)(\sum_{l=1}^{|L|} \sum_{a \epsilon A''_l} w_a)/2 = ((T-1)/2) \sum_{a \epsilon LPG} w_a$, and $W_{\sigma'}$ satisfies the Assignment bound.

4 CONCLUSION

Algorithms for scheduling priority-slots so as to obtain low values of session schedule-delays have been presented, following an NP-completeness proof showing that the minimum-delay scheduling problem is algorithmically hard for general networks. The network model of Section 1.2 is used, along with the assumption that the paths for all sessions in the network are known.

Algorithm A_{tree} constructs a schedule with sum of schedule-waits equal to zero, for networks with tree link-precedence graphs. For such networks, this algorithm can be extended so as to produce schedules that have waits between consecutive slots equal to known and fixed link propagation or nodal switching times.

Algorithm $A_{heuristic}$ is a scheduling heuristic that can be used for any network that conforms to the model assumed. This algorithm reduces to A_{tree} for tree link-precedence graphs. An analysis of its worst-case performance has been presented, but its performance for specific networks may need to be investigated further. $A_{heuristic}$ can be extended to the non-zero link propagation or nodal switching time case, and also to the case of different slot durations on different links.

The schedule may require improvement during the course of operation of the network, as new sessions are set up and old ones disconnected. In order to reduce the time and communication required, the schedule may be computed using a distributed algorithm, where the computation and communication involving a link-precedence graph vertex are the joint responsibility of the transmitting and receiving nodes for the corresponding link.

A distributed implementation of algorithm $A_{heuristic}$ is possible. The distributed algorithm described by Gallager, Humblet, and Spira [2] would be used to compute a maximum-weight spanning tree and corresponding root vertex for each link-precedence graph component. Then, links would be scheduled using communication outwards along the link-precedence trees starting from the roots.

In order to improve the schedule, links may be rescheduled using an optimal assignment algorithm, holding fixed the priority-slot assignments on neighbouring links. The distribution of rescheduling opportunities among links can be done in the manner of the solution described by Chandy and Misra [1] for the distributed dining philosophers problem.

5 ACKNOWLEDGEMENTS

I wish to thank Professor Robert Gallager for having supervised my thesis research on which this paper is based, and for reviewing this paper. This research was conducted at the MIT Laboratory for Information and Decision Systems, with support provided by the Defence Advanced Research Projects Agency under Contract ONR/N00014-84-K-0357 and the National Science Foundation under Grant NSF-ECS-8310698.

6 REFERENCES

[1] Chandy, K.M., and J. Misra, "The Drinking Philosophers Problem," *ACM Transactions on Programming Languages and Systems,* Vol. 6, No. 4, October 1984, pp. 632-646.

[2] Gallager, R.G., P.A. Humblet, and P.M. Spira, "A Distributed Algorithm for Minimum-Weight Spanning Trees," *ACM Transactions on Programming Languages and Systems,* Vol. 5, No. 1, January 1983, pp. 66-77.

[3] Garey, M.R., and D.S. Johnson, *Computers and Intractability (A Guide to the Theory of NP-Completeness),* W.H. Freeman and Co., New York, 1979.

[4] Graves, S.C., H.C. Meal, D. Stefek, and A.H. Zeghmi, "Scheduling of Re-entrant Flow Shops," *Working Paper*, Sloan School of Management, MIT, Cambridge, MA, November 23, 1982.

[5] Hajek, B.E., private communication, January 1986.

[6] Hitz, K.L., "Scheduling of Flexible Flow Shops," *Laboratory for Information and Decision Systems Report 879*, MIT, Cambridge, MA, March 1979.

[7] Kogge, P.M., *The Architecture of Pipelined Computers,* Hemisphere Publishing Corporation, McGraw-Hill, New York, 1981.

[8] Mukherji, U., "A Schedule-Based Approach for Flow-Control in Data Communication Networks," *Laboratory for Information and Decision Systems Report 1527,* Sc.D. Thesis Dissertation, MIT, Cambridge, MA, January 1986; also, to be presented at IEEE Global Telecommunications Conference in December 1986.

[9] Orlin, J.B., "The Complexity of Dynamic Languages and Dynamic Optimization Problems," *Proceedings ACM STOC*, Milwaukee, 1981, pp. 218-227.

[10] Papadimitriou, C.H., and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity,* Prentice-Hall, Englewood Cliffs, N.J., 1982.

The Algebraic and Geometric Structure of Markovian Petri Network Lattices

Aurel A. Lazar and Thomas G. Robertazzi
Columbia University and SUNY at Stony Brook
Departments of Electrical Engineering

ABSTRACT

A class of Markovian (Stochastic) Petri Net models whose equilibrium probabilities exhibit local balance and have a Jackson type product form solution is presented. The natural topological space for embedding the state transition lattices of this class is shown to be a multidimensional torroidal manifold. Examples include a dual processor bus architecture and a station oriented CSMA model.

I. Introduction

Petri Network models originated from the work of C.A. Petri in 1962 [16]. They are useful for modeling and graphically illustrating concurrency, serializability, synchronization and resource sharing in complex systems [15]. Recently attention has been given to including appropriate timing mechanisms in Petri Net models. Stochastic Petri Nets (SPN's) [13] are based on the assumption that the time between successive events can be modeled as a random variable. To date most such work has had its basis in Markovian statistics. We will refer to such Stochastic Petri Nets as Markovian Petri Nets (MPN's).

The use of Petri Net models can become difficult as the complexity of systems increase. This has generally been attributed to a perceived unwieldiness of the Petri Net schematic representation and the potentially large state transition lattices (diagrams) associated with it. In fact, the essential problem is not a large state transition lattice per se but rather the lack of a known structure in the lattice. As an example, the lattices of certain classes of queueing network models [1], [4], [6] exhibit enough *structure* [7], [8], [9], [18] so that simple algebraic product form solutions exist for the equilibrium state probabilities. Yet, the lattice can be arbitrarily large or even infinite in extent.

In this paper a large class of Markovian Petri Nets is introduced whose equilibrium state probabilities have a product form solution and which also satisfy a type of local balance. Moreover the lattices possess an elegant topological structure. They are most naturally embedded in multidimensional torroidal manifolds and so will be referred to as "Torroidal Markovian Petri Networks". The topological structure also suggests means of enhancing the clarity of the Petri Net schematic.

This paper is organized as follows. In section II an example of a dual processor architecture with bus interconnection is discussed. The general algebraic topological structure of this class is presented in section III. A station oriented model of the CSMA protocol is examined in section IV in relation to torroidal MPN's.

II. A Multiprocessor Model with Bus Oriented Interconnection.

Consider a dual processor computer system as shown in Fig. 1. It consists of two processors, P_1 and P_2, and a common shared memory, CM. P_1 or P_2, or both, may attempt transfers to CM through the bus. Conflicts occur as only one processor may utilize the bus, and hence access memory CM, at one time.

A Markovian Petri Net model of this system appears in Fig. 2. Formally, the MPN is defined as the six-tuple:

$$\mathcal{P} = (P, T, I, O, M, Q) \quad . \tag{2.1}$$

P is a set of places (drawn as circles) and T is a set of transitions (drawn as horizontal bars). I is an input function which maps each transition to one or more places (drawn as directed arcs) and O is an output function which maps each place to one or more transitions (drawn as directed arcs). A marking, M, is an assignment of a non-negative number of tokens (drawn as dots) to individual places. For a given Petri Net, the marking defines the state of the system. Q is the set of rates associated with the transitions.

The rules of Petri Net operation are simple. If there is at least one token in each place incident to a transition, it is enabled for firing. This involves removing one token from each place incident to the transition and adding one token to each place connected to the transition by an outgoing arc. For a MPN the time between an enabled transition and its firing is an exponentially distributed random variable.

The class of MPN's considered in this paper is safe, that is, a place may hold at most one token at any given time. All transitions also exhibit liveness, *i.e.*, deadlock cannot occur.

In the dual processor MPN model of Fig. 2 one can see a total of two linear sequences of events originating from the places indicating that the processors are internally active (externally idle). Each such sequence will be referred to as a *task sequence* and the states which comprise it as *sub-tasks.* The first state is often an idle state (*i.e.*, task waiting to be initiated). The two tasks of Fig. 2 thus correspond to P_1 or P_2 attempting to perform a transfer with CM.

Generally, one can distinguish between places which represent subtasks and those which represent resources (*e.g.*, BUS FREE). The essence of the Petri Net paradigm is that a task sequence may be blocked (temporarily or permanently) if resources are not available.

In Fig. 2, q^{kl} is the exponential rate associated with the l th sub-task (l = 0,1,2) of the k th task sequence (k=1,2). The Petri Net transitions should not be confused with the transitions of the state transition lattice of the Markov process associated with the Petri Net. The corresponding state transition lattice appears in Fig. 3. The horizontal axis corresponds to P_1 and task sequence 1 while the vertical axis corresponds to P_2 and task sequence 2. The equilibrium state probabilities, for each quadrant of the lattice, are given below for all states (i, j) of the indicated task sequences:

$$p(i,j) = \frac{q^{10}}{q^{1i}} \frac{q^{20}}{q^{2j}} p(0,0) \quad . \tag{2.2}$$

Note that these probabilities have the characteristic product form which is known to exist for certain classes of queueing networks. This is a surprising result whose basis will be explained in the next section in terms of the algebraic topological structure of the lattice of the Markov chain associated with the MPN.

III. Toroidal MPN Lattices

To understand why the MPN of Fig. 2 has a product form solution, consider two *independent* task sequences. That is, sufficient resources exist so that the state of each sequence does not affect the state of the other sequence. This leads to the state transition lattice of Fig. 4. Let task 1 correspond to the horizontal axis and task 2 correspond to the vertical axis.

While the position of the states appears to conform to a rectangular geometry, when one considers the "wrap around" character of the boundary transitions it becomes apparent that the natural topological space for embedding the lattice is the surface of a torus. The class of MPNs that can be embedded on a torus will be referred to as *Toroidal Markovian Petri Networks.*

The torus, a two dimensional manifold, is a smooth surface without boundary. Therefore, the global balance equations at each point in the state transition diagram are structurally the same. Boundary states, which often appear in queueing based lattices, need not be considered separately.

While any state may be chosen as a reference state, in applied terms the natural choice is the state which corresponds to both independent tasks being idle. With this state being given by the coordinates (0,0), we seek an expression for equilibrium probabilities $p(i,j)$. The global balance equation for the conservation of probability flux at vertex (i,j) is:

$$(q^{1i} + q^{2j})\, p(i,j) = q^{1,i-1} p(i-1,j) + q^{2,j-1} p(i,j-1) \quad . \tag{3.1}$$

Previous work [7], [8], [9], [18] on the algebraic topological structure of queueing network lattices has shown that the classical Jackson-type product form solution of the equilibrium probabilities can be traced back to a decomposition of the state transition diagram into elementary building blocks. For a class of lattices [18], cyclic flows can be associted with these building blocks. A cyclic flow is associated with the transitions of each building block and has equal magnitude of probability flux on each transition. Note that, the lattice structure in Fig. 4 suggests the presence of cyclic flows of transitions about the torus (*e.g.*, (0,0), (1,0), (2,0), (3,0), (4,0) to (0,0)). It will be shown that these cycles are the fundamental building blocks [7] of the lattice.

This observation is equivalent (Duality Principle [18]) to the hypothesis that the following local balance equations are satisfied :

$$q^{1,i-1} p(i-1,j) = q^{1i} p(i,j) \quad , \tag{3.2}$$

$$q^{2,j-1} p(i,j-1) = q^{2j} p(i,j) \quad .$$

These local balance equations are suggestive of a natural recursion for $p(i,j)$:

$$p(i,j) = \frac{q^{10}}{q^{1i}}\frac{q^{20}}{q^{2j}}p(0,0) \quad . \tag{3.3}$$

This does indeed satisfy the global balance equations (3.1). For N independent tasks the expression can be generalized in a straight forward manner to :

$$P(k_1,k_2,...,k_N) = \prod_{l=1}^{N}\frac{q^{l0}}{q^{lk_l}}p(0,0,...,0) \quad , \tag{3.4}$$

where k_l is the k_l th sub-task of the l th task sequence. That we have achieved a product form solution is not surprising at this point since task independence was assumed.

Note that a torus is a two dimensional space, even though it is often visualized in terms of a doughnut shaped model in three dimensional space. An equally valid representation is as a rectangular area with opposite edges identified as being equivalent. This identification corresponds to the "wrap around" of Fig. 4. For any non-negative N in (3.4), there is a corresponding N dimensional torroidal manifold in which the lattice may be embedded. For instance, if N=3 the 3-manifold may be represented as a cube in R^3 with opposite faces identified as being equivalent.

Note that the lattice of Fig. 3 can be seen to have the same structure as the lattice of Fig. 4 *with specific cyclic building blocks removed.* The removal of a building block(s) does not affect the form of the solution, except to cause a renormalization of p(0,0,...,0). Consistency of the local balance equations is preserved since a form of geometric replication applies [7], [9]. Therefore, the flow on a cycle is isolated [18], *i.e.*, it neither contributes nor removes probability flux from the other parts of the lattice.

The lattice of the dual processor system embedded in a torus in R^3 is shown in Fig. 5. It may seem that there is something intrinsically different about the orientations of the isolated cyclic flows of task sequence 1 and 2. From Fig. 2 and Fig. 3 it can be seen that the orientations should indeed be symmetrical. The *apparent* difference occurs because of the visualization of the manifold in R^3.

The properties of the MPN discussed above have been characterized in terms of the algebraic topological structure of the state transition diagram of the associated Markov chain. This characterization can be translated into the Petri Net schematic representation as follows. Consider a Markovian Petri Net consisting of a number of tasks that comprise a series of sequential sub-tasks. A task is only allowed to commence if there is a non-zero probability that it can be completed without the need for a state change in other task sequences. If this characterization does not hold, the simple product form solution may not be valid.

IV. A Station Oriented CSMA Protocol Model

Since bus oriented multiprocessor architectures can be modeled by Petri Nets, the modeling of bus type local area networks is another potential application. Petri Net models of bus type local area networks have appeared in [5], [14].

Bus type local area network models may either attempt to model the various stations and their interaction in detail or to model the behavior of a single station in the context of the aggregate statistical behavior of the remainder of the network. A CSMA model of the latter type will now be introduced. It should be noted that the difficulty in taking the former approach for local area networks, as opposed to multiprocessor bus interconnections, lies in the non-negligible propagation delay along the bus.

A MPN of the LAN station appears in Fig. 6 and the corresponding lattice in Fig. 7. Messages originate at the station with rate q^{10}. They are transmitted after an average time of $1/q^{11}$ spent in channel scanning. Once transmission has been initiated, collisions occur with probability ϵ after an average time of $1/q^{12}$. The average duration of a successful transmission is $1/q^{12} + 1/q^{13}$. The average collision resolution time is τ.

Note that the q^{11} transition in the Petri Net does not depend on the bus availability. While, bus availability could be modeled stochastically, more correctly it depends on the states of other stations and their connection through the bus with significant propagation delay. The simplified model of Fig. 7 will be used to illustrate some relevant ideas concerning the torroidal class rather than as a detailed representation of the system operating under the CSMA protocol.

The equilibrium probabilities of the states along the lattice's main loop are:

$$p(k) = \frac{q^{10}}{\alpha q^{1k}} p(0) \quad , \tag{4.1}$$

where α is $(1-\epsilon)$ when k represents the MESSAGE READY and START TRANSMIT states and 1 otherwise. One way of viewing the lattice of Fig. 7 is as a cycle embedded in a one dimensional torroidal manifold with a "feedback" loop attached which corresponds to the possibility of collisions. The product form is maintained since the split of probability flux between the two loops is proportional to $1-\epsilon$ and ϵ.

However, when searching for a natural topological space for embedding the entire lattice must be considered. While this lattice can be embedded on a 2-d torus, this could also be accomplished using a 2-d spherical manifold. A circle is a one dimensional torroidal manifold as well as a degenerated one dimensional spherical manifold.

The key point in the analysis above is the random bifurcation of flows [10] (see Fig. 6 and 7). Thus it would seem most appropriate to view the lattice as being embedded on a manifold obtained by the "pasting together" (an algebraic topological concept) of two 1-d torroidal manifolds. The pasting occurs along shared transitions and has been previously noted for queueing networks with random routing [18]. A detailed example of pasting along nodes appears in [10].

V. Conclusion

A class of Markovian Petri Net models with equilibrium probabilities which can be expressed in the product form and satisfy local balance equations has been described. Specific models have been presented which are practical examples of resource sharing and protocol operation.

Somewhat surprising to us (in particular with regards to the well known dining philosopher's problem [10]) the equilibrium probabilities can be calculated by means of simple analytical expressions. The results required investigating the algebraic, geometric and topological properties of the relevant state transition lattices. The structure of state transition lattices has been unraveled and understood using a simple and intuitive approach.

The wide spread use of Petri Net models has been somewhat inhibited by a perceived unwieldiness of the Petri Net schematic. The authors have found that drawing the task sequences in a linear form and not completely drawing directed arcs to and from resource places tends to enhance the clarity and reliability of Petri Net schematics. While such a structured approach has sometimes been implicit in previous work, this research provides an explicit basis for its use.

Acknowledgements

The authors would like to acknowledge useful discussions with Prof. C.C. Yeh. This paper has also benefited from the illustrative skills of L. Koh. The research reported in this paper was supported in part by the National Science Foundation under Grant CDR-44-21402 and in part by the Office of Naval Research under Contract N00014-85-K0610.

References

[1] Baskett, F., Chandy, K.M, Muntz, R.R, and Palacios, F., "Open, Closed and Mixed Networks of Queues with Different Classes of Customers," *JACM*, Vol. 22, No. 2, April 1975, pp. 248-260.

[2] Dijkstra, E.W., "Cooperating Sequential Processes," *Programming Languages*, F. Genvys (editor), Academic Press, New York, 1968, pp. 43-112.

[3] Ghanta, S. and Bruell, S.C., "Throughput Bounds for Generalized Stochastic Petri Net Models," *International Workshop on Timed Petri Nets*, July 1985, Torino Italy, IEEE Comp. Soc. Press, 1986, pp. 250-262.

[4] Gordon, W.J. and Newell, G.F., "Closed Queueing Systems with Exponential Servers," *Operations Research* 15, No.2, 1967, pp. 254-265.

[5] Gressier, E., "A Stochastic Petri Net Model for Ethernet," *International Workshop on Timed Petri Nets*, July 1985, Torino, Italy, IEEE Comp. Soc. Press, 1986, pp. 296-306.

[6] Jackson, J.R., "Networks of Waiting Lines," *Operations Research*, No.5, 1957.

[7] Lazar, A.A. and Robertazzi, T.G., "The Geometry of Lattices for Markovian Queueing Networks," *Research Report, Center for Telecommunications Research,* 1984, submitted for publication.

[8] Lazar, A.A. and Robertazzi, T.G., "The Geometry of Lattices for Multiclass Markovian Queueing Networks," *Proceedings of the Eighteenth Conference on Information Sciences and Systems,* Priceton University, Princeton, NJ, 1984, pp. 164-168.

[9] Lazar, A.A., "An Algebraic Topological Approach to Markovian Queueing Networks", *Proceedings of the Eighteenth Conference on Information Sciences and Systems,* Priceton University, Princeton, NJ, 1984, pp. 437-442.

[10] Lazar, A.A. and Robertazzi, T.G., "Markovian Petri Net Protocol Models with Product Form Solution", SUNY at Stony Brook, CEAS Tech. Rep. 480, 1986.

[11] Marsan, M.J., Balbo, G., Conte, G. and Gregoretti, F., "Modeling Bus Contention and Memory Interference in a Multiprocessor System," *IEEE Trans. on Comp.,* Vol. C-32, No.1, Jan 1983, pp. 60-72.

[12] Massey, W., *Algebraic Topology: an Introduction,* Springer Verlag, New York, 1977.

[13] Molloy, M.K., "Performance Analysis Using Stochastic Petri Nets," *IEEE Trans. on Comp.,* Vol C-31, No.9, Sept 1982, pp. 913-917.

[14] Molloy, M.K., "Modeling and Analysis of LAN Protocols Using Labeled Petrinets", TR-84-15, Dept. of Computer Science, The University of Texas at Austin, Austin, TX, Sept. 1984.

[15] Peterson, J.L., *Petri Net Theory and the Modeling of Systems,* Prentice- Hall, 1981.

[16] Petri, C.A., *Kommunikation mit Automaten,* Ph.D. Thesis, Univ. of Bonn, Fed. Rep. of Germany, 1962.

[17] Shapiro, S.D., "A Stochastic Petri Net with Applications to Modeling Occupancy Times for Concurrent Task Systems," *Networks,* Vol. 9, pp. 375-379, 1979.

[18] Wang, I.Y. and Robertazzi, T.G., "The Probability Flux Circulation of Certain Communication and Computation State Models," *Proc. of 1986 Information Sciences and Systems Conference,* Princeton, NJ, March 1986. Extended version: CEAS Tech. Rep. 477, 1986, SUNY at Stony Brook, submitted for publication.

[19] Zenie, A., "Colored Stochastic Petri Nets," *International Workshop on Timed Petri Nets,* July 1985, Torino, Italy, IEEE Comp. Soc. Press, 1986, pp. 262-271.

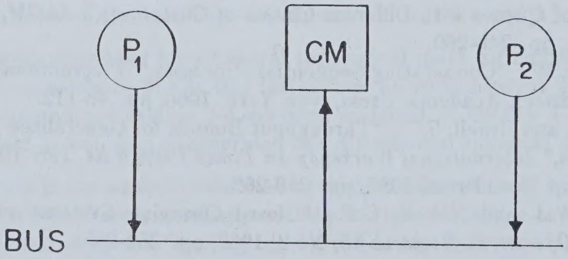


FIGURE 1

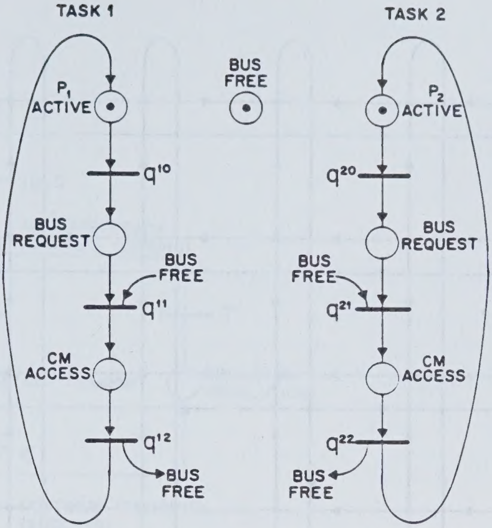

TASK 1
TASK 2
BUS FREE
P1 ACTIVE
P2 ACTIVE
q10
q20
BUS REQUEST
BUS REQUEST
BUS FREE
BUS FREE
q11
q21
CM ACCESS
CM ACCESS
q12
q22
BUS FREE
BUS FREE

FIGURE 2

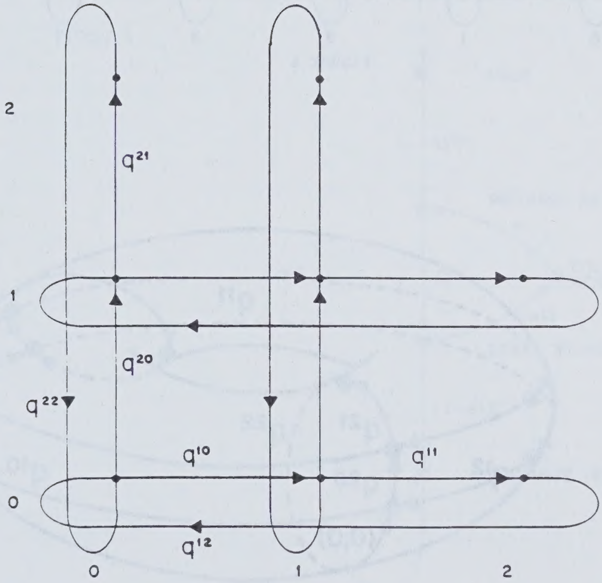

2
1
0
q21
q20
q22
q10
q11
q12
0
1
2

FIGURE 3

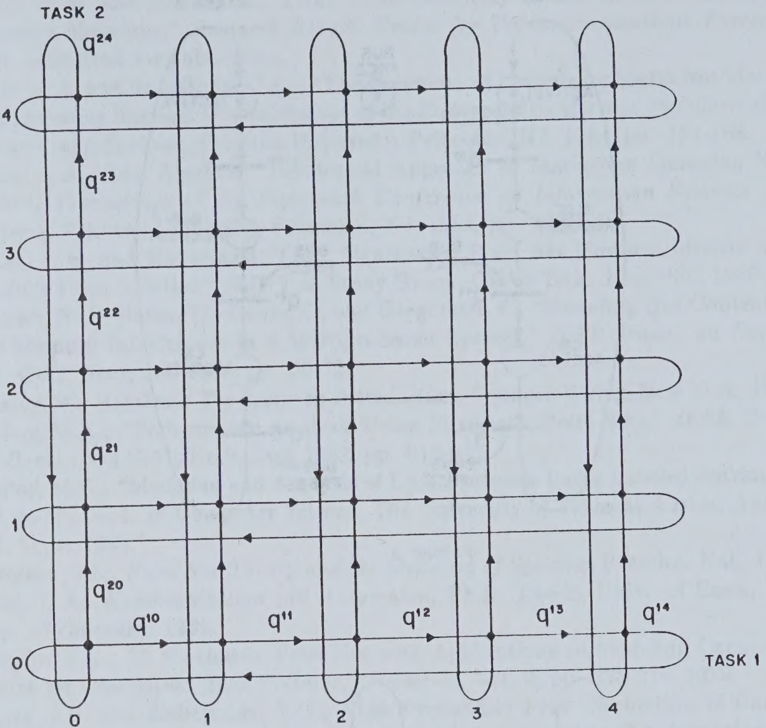

TASK 2
q24
4
q23
3
q22
2
q21
1
q20
0
q10
q11
q12
q13
q14
TASK 1
0
1
2
3
4

FIGURE 4

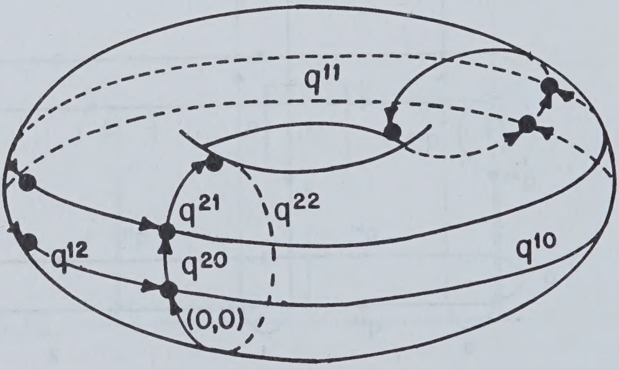

q11
q21
q22
q12
q20
q10
(0,0)

FIGURE 5

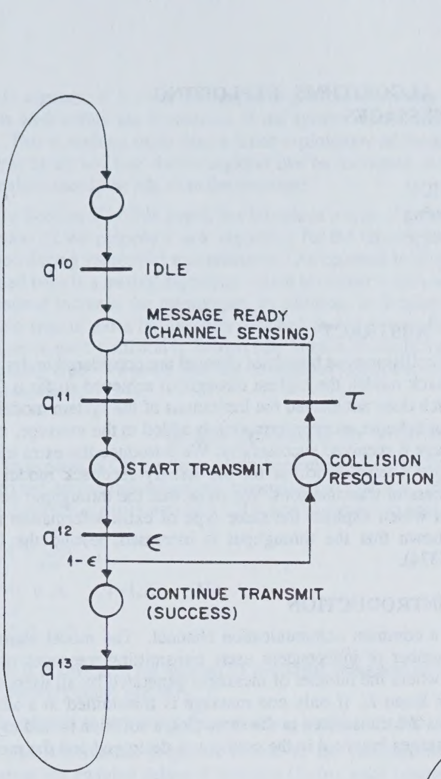

q^{10}
IDLE
MESSAGE READY
(CHANNEL SENSING)
q^{11}
τ
START TRANSMIT
COLLISION
RESOLUTION
q^{12}
ϵ
$1-\epsilon$
CONTINUE TRANSMIT
(SUCCESS)
q^{13}

FIGURE 6

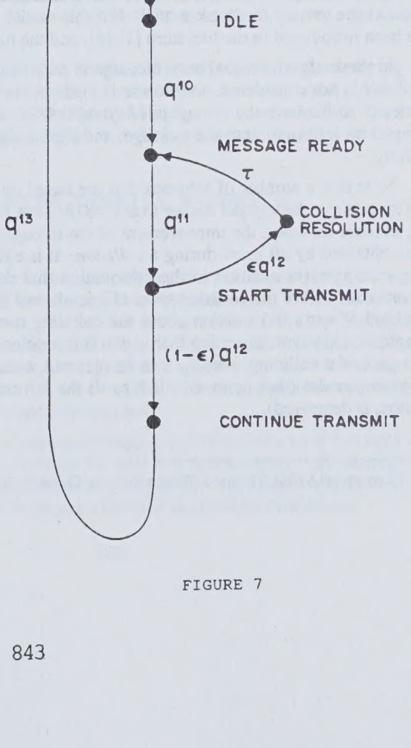

IDLE
q^{10}
MESSAGE READY
τ
q^{13}
q^{11}
COLLISION
RESOLUTION
ϵq^{12}
START TRANSMIT
$(1-\epsilon)\, q^{12}$
CONTINUE TRANSMIT

FIGURE 7

MIXING COLLISION RESOLUTION ALGORITHMS EXPLOITING INFORMATION OF SUCCESSFUL MESSAGES

ILAN KESSLER and MOSHE SIDI[1]
Department of Electrical Engineering
Technion -Israel Inst. of Technology
Haifa 32000, Israel.

ABSTRACT

Algorithms for accessing a slotted-time collision-type broadcast channel are considered under a Poisson infinite-user model. For the ternary feedback model, the highest throughput achieved so far is 0.4878. In this paper, we introduce a new scheme which does not exceed the limitations of the system model, yet it achieves throughput higher than 0.4878. In our scheme, extra information is added to the message, and this information can be used only when the message is received successfully. We introduce the extra information to be added to the message, and propose a new algorithm for the ternary feedback model which exploits this information obtained during successful transmissions. We show that the throughput achieved is 0.4892. We also propose a new algorithm which exploits the same type of extra information for the known-multiplicity feedback model. It is shown that the throughput is increased beyond the highest throughput achieved so far for this model (0.5324).

I. INTRODUCTION

We consider algorithms for accessing a common communication channel. The model assumed is commonly used in the literature - infinite number of independent users transmitting messages of equal length over a slotted-time broadcast channel, where the number of messages generated by all users collectively in each slot is Poisson distributed with mean λ; if only one message is transmitted in a slot, it is received successfully; if two or more messages are transmitted in the same slot, a collision is said to occur, and all the information included in all the messages involved in the collision is destroyed and the messages must be retransmitted at some later time.

The most common situation is that each user observes the channel and determines at the end of each slot whether that slot was idle, or there was a successful transmission, or a collision has occurred. This is known as the ternary feedback model. For this model, a number of collision resolution algorithms (CRA) have been introduced in the literature [1]-[6], and the highest throughput achieved so far is 0.4878 [6].

In these algorithms, when a message is received successfully, the information included in the message itself is not considered, and no use is made of the fact that this information is available. However, in an attempt to increase the throughput beyond 0.4878, one may think of a scheme that utilizes this fact by adding extra information to the message, and exploiting this information when the message is received successfully.

Note that a number of schemes that are based on exploiting extra information added to the message, and that achieve throughput higher than 0.4878, have been introduced in the literature (e.g. [7]-[9]). However, in these schemes, the improvement of the throughput is due to the assumption that this extra information is obtained by all users during a *collision.* It is easy to realize, that if extra information about the colliding messages (in addition to the information that there was a collision) is obtained during a collision, then one can resolve the collision more efficiently and thus increase the throughput. It is also easy to realize what kind of extra information about the colliding messages should be obtained in order to increase the throughput. However, assuming that useful information can be obtained during a collision - in other words, that a part of a colliding message can be restored without errors - exceeds substantially the limitations of the system model (that upon a collision, all the information included in all the messages involved in the collision, is destroyed).

---

[1] Currently with IBM, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598

In contrast, it is clear that exploiting information that is included in a successfully transmitted message, is well within the limitations of the system model, since such information is anyway available to all users. This is nothing more than a fuller exploitation of the available features of the system. However, it is not clear at all whether the throughput can be increased in such a way. It is also not clear what kind of information should be added to the message.

In Section II of this paper, we introduce a type of information that can be added to the message, and in Section III we propose a new algorithm for the ternary feedback model, which exploits this information obtained during successful transmissions. As opposed to all previously presented algorithms, the algorithm proposed here is a *mixing* algorithm which is neither a DIA nor an FCFSA [10]. We show that this scheme does indeed increase the throughput. In addition, in Section IV we propose a new algorithm that exploits the same type of extra information obtained during successful transmissions, when the number of colliding messages in each collision is known (the known-multiplicity feedback model [11]). This algorithm is also of a mixing type. We show that the throughput is increased beyond the highest throughput achieved so far for this model (0.5324).

II. SET MARKER IN MIXING ALGORITHMS

In general, a CRA selects at the beginning of each slot a subset A of the time axis, and all messages that arrived during this subset are transmitted in that slot. This is referred to as *enabling* the subset A. We define a *mixing algorithm* to be a CRA that at some step enables a subset A that satisfies:

1) $A = \bigcup_{i=1}^{N} A'_i$;

2) $A'_i \in A_i \qquad i=1,2,\cdots,N$;

3) $\bigcap_{i=1}^{N} A_i = \phi$;

4) It is known that the number of messages contained in A_i is greater than zero, for all $1 \le i \le N$, except possibly some single i_0.

This is referred to as mixing the subsets A_i $i=1,2,\cdots,N$.

The information proposed to add to the message refers to mixing algorithms and is as follows: Whenever the enabled subset A satisfies (1)-(4), each transmitted message contains the index j of the subset A_j $(1 \le j \le N)$ to which the message belongs (this requires $\lceil \log_2 N \rceil$ bits). In particular, in this work we propose mixing algorithms that mix only two subsets A_1 and A_2, and the proposed information to be added to the message is then a single bit that indicates to which of the two subsets, A_1 or A_2, the message belongs. We refer to this bit as *set marker*.

The following notations are used in the sequel: The number of messages that arrived during some subset A of the time axis is denoted by $[A]$. If $[A]=1$, and A is a mix of the subsets A_1 and A_2, and the single message arrived during A belongs to A_1, then it is denoted by $[A_1 \cup A_2]=1A_1$ (and similarly if the message belongs to A_2). A subset A for which $[A]\ge 2$ is called a *collision interval.* A subset A for which $[A]=k$, is called an *interval of multiplicity* k.

III. AN ALGORITHM FOR THE TERNARY FEEDBACK

A. Operation of the Algorithm - General Description

Consider two time axes. The first, called the arrival axis, shows the Poisson arrival instants. The second, called the transmission axis, is segmented into consecutive intervals called *Collision Resolution Intervals (CRI)*, each consisting of integral number of slots. At the first slot of the i-th *CRI*, the algorithm enables a subset E_i of the arrival axis. If no collision occurs in this slot, the i-th *CRI* terminates and a new *CRI* starts immediately thereafter. If a collision occurs, the algorithm initiates a *Collision Resolution Process (CRP)* whose termination defines the termination of the i-th *CRI*. Next, the algorithm enables a new subset E_{i+1}, and so on. In this way the arrival axis is entirely searched.

At the end of each *CRI*, the arrival axis can be segmented into disjoint intervals, such that each interval belongs to one of the following categories: 1. Interval for which it is known that all messages that arrived during it have already been successfully transmitted; 2. Interval for which nothing is known about the number of messages that arrived during it, beyond the knowledge of the Poisson distribution.

Definition:
The *New Intervals Pool (NIP)* at the beginning of some slot of the i-th *CRI*, is the continuous and ordered (according to increasing time index) union of:
all intervals of category (2) at the end of the $(i-1)$-st *CRI*, minus certain intervals as determined by the algorithm from the beginning of the i-th *CRI* until that moment.

A *new interval* is defined to be a continuous interval of the *NIP*, starting at the initial instant of the *NIP*. E_i is a new interval whose length is x (where x is a parameter to be optimized for achieving maximum throughput). In Fig. 1 a typical picture of the arrival axis at the end of a *CRI* is shown, and the selection of E_i is demonstrated.

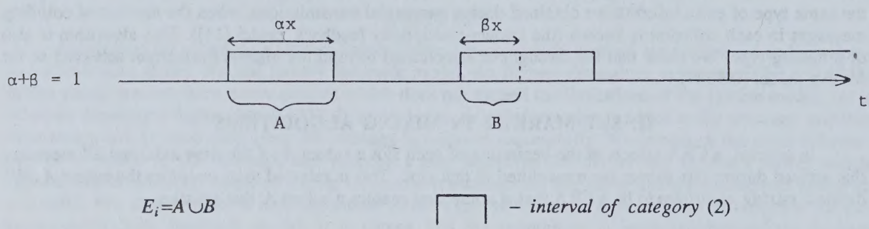


Fig. 1: The arrival axis at the end of a *CRI* (typically), and the selection of E_i.

B. The Collision Resolution Process

Before describing the *CRP* formally, it will be helpful to give an informal description. Referring to Fig. 2, E is the collision interval to be resolved. In the first step, the process enables the subinterval E' together with F which is a new interval.

- If the outcome is $[E' \cup F]=0$, then there is no need to treat E' and F any longer (all messages contained in them have been successfully transmitted). It is also clear now that $[E-E'] \geq 2$, and the process is repeated for $(E-E')$.
- If the outcome is $[E' \cup F]=1F$, that is, the transmission was successful and the set marker indicated that the message belongs to F (i.e. $[E']=0$ and $[F]=1$), then the situation is similar to that of the previous case and the same action is taken.
- If the outcome is $[E' \cup F]=1E'$, then the situation is again similar, but now $[E-E'] \geq 1$ and the next step is to enable this interval – if $[E-E']=1$ then the process terminates, and if $[E-E'] \geq 2$ then the process is repeated for this interval.
- If the outcome is $[E' \cup F] \geq 2$, then in the next step the process enables E'. If $[E']=0$, then there is no need to treat this interval any longer; it is also clear now that $[E-E'] \geq 2$ and $F \geq 2$, and the process is repeated for each of these intervals. If $[E']=1$, then there is no need to treat this interval any longer; it is also clear now that $[E-E'] \geq 1$ and $F \geq 1$, and each of these intervals is enabled – if it contains only one message then it is not treated any longer, otherwise the process is repeated for it. If $[E'] \geq 2$, then it is easy to see that now nothing is known about the number of messages contained in each of the intervals $(E-E')$ and F, beyond the knowledge of the Poisson distribution ,and both are merged into the *NIP*; the process is repeated for E'.

We now give a formal description of the *CRP*. The process accepts in its input a subset E of the arrival axis, which is known to be a collision interval. During the process, additional intervals each of which is known to be a collision interval, are generated. The process appends, and later on deletes, some of these intervals to a list named *Collision Intervals List (CIL)* (in a FIFO manner), as specified in the *CRP* description. At the beginning and at the end of the process, the *CIL* is empty. The process also deletes and/or merges certain intervals to the *NIP*. While *merging* intervals is specified explicitly in the *CRP*

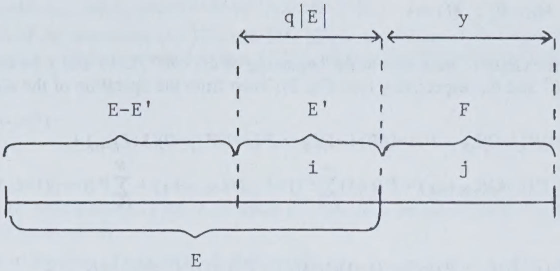


Fig. 2: Mixing a collision interval with a new interval.

description, *deleting* is done according to the following rule:
Rule: A new interval which is contained in an enabled subset in some slot, is deleted from the *NIP* at the beginning of that slot.
The process is defined by the following sequence of steps.

The Collision Resolution Process

```
(1) Enable a right-hand subinterval $E'$ of $E$ whose length is $|E'|=q|E|$ $(0<q<1)$ together with a new interval $F$ whose length is $y$,
        If $[E' \cup F]=\{0 \text{ or } 1F\}$: Denote $(E-E')$ by $E$ and goto (1)
        If $[E' \cup F]=1E'$: Denote $(E-E')$ by $E''$ and goto (5)
        If $[E' \cup F]\geq 2$: Goto (2)
(2) Enable $E'$,
        If $[E']=0$: Append $F$ to the CIL; denote $(E-E')$ by $E$ and goto (1)
        If $[E']=1$: Goto (3)
        If $[E']\geq 2$: Merge $(E-E')$ and $F$ into the NIP; denote $E'$ by $E$ and goto (1)
(3) Enable $(E-E')$,
        If $[E-E']=1$: Denote $F$ by $E''$ and goto (5)
        If $[E-E']\geq 2$: Denote $(E-E')$ by $E$ and goto (4)
(4) Enable $F$,
        If $[F]=1$: Goto (1)
        If $[F]\geq 2$: Append $F$ to the CIL and goto (1)
(5) Enable $E''$,
        If $[E'']=1$: Goto (6)
        If $[E'']\geq 2$: Denote $E''$ by $E$ and goto (1)
(6) Check the CIL:
        If it is empty: Terminate
        Otherwise: Take the next interval from the CIL, denote it by $E$ and goto (1)
```

C. Performance Analysis

Let L_N denote the mean length of a *CRI* that exactly N messages were transmitted in its first slot, and let M_N denote the mean number of successfully transmitted messages during that *CRI*. From the operation of the algorithm it is clear that:

$$L_0=L_1=1 \quad ; \quad M_0=0 \quad ; \quad M_1=1 \tag{3.1}$$

For $N \geq 2$, E contains exactly N messages at the beginning of the *CRP*. Let i and j be the number of messages contained in E' and F, respectively (see Fig. 2). Then from the operation of the algorithm it follows that for $N \geq 2$:

$$\begin{aligned} L_N = {} & 1 + P(i=0)P(j=0)L_N + P(i=0)P(j=1)L_N + P(i=1)P(j=0)(1+L_{N-1}) + \\ & + P(i=0)\sum_{k=2}^{\infty}P(j=k)(L_N+L_k) + P(i=1)\sum_{k=1}^{\infty}P(j=k)(2+L_{N-1}+L_k) + \sum_{r=2}^{N}P(i=r)(1+L_r) \end{aligned} \tag{3.2}$$

$$\begin{aligned} M_N = {} & P(i=0)P(j=0)M_N + P(i=0)P(j=1)(1+M_N) + P(i=1)P(j=0)(1+M_{N-1}) + \\ & + P(i=0)\sum_{k=2}^{\infty}P(j=k)(M_N+M_k) + P(i=1)\sum_{k=1}^{\infty}P(j=k)(1+M_{N-1}+M_k) + \sum_{r=2}^{N}P(i=r)M_r \end{aligned} \tag{3.3}$$

where

$$P(i=r) = \binom{N}{r} q^r (1-q)^{N-r} \qquad r=0,1,\cdots,N \tag{3.4a}$$

$$P(j=k) = \frac{e^{-\lambda y}(\lambda y)^k}{k!} \triangleq \frac{e^{-z}z^k}{k!} \qquad k \geq 0 \tag{3.4b}$$

(3.2) is explained as follows: The 1 is the first slot of the *CRI*, in which the initial collision is detected. When $i=j=0$ or $i=0$ and $j=1$, the *CRI* lasts $1+(L_N-1)$ slots more on average - one slot to detect i and j, plus L_N-1 slots on average to resolve the collision among the N messages contained in $(E-E')$ (one slot less since the collision is known in advance). When $i=1$ and $j=0$, the *CRI* lasts $1+L_{N-1}$ slots more on average - one slot to detect i and j, plus L_{N-1} slots on average to resolve the $N-1$ messages contained in $(E-E')$ ($N-1 \geq 1$ and therefore it is required to enable $(E-E')$ itself, too). When $i=0$ and $j=k \geq 2$, the *CRI* lasts $2+(L_N-1)+(L_k-1)$ slots more on average - two slots to detect that $i+j \geq 2$ and $i=0$, plus L_N-1 and L_k-1 slots on average to resolve the collisions (which are known in advance) among the N messages contained in $(E-E')$ and the k messages contained in F, respectively. The case when $i=1$ and $j=k \geq 1$ is explained similarly, except that now it is required to enable the intervals themselves before resolving them, as in the case when $i=1$ and $j=0$. Finally, when $i=r \geq 2$, the *CRI* lasts $2+(L_r-1)$ slots more on average - two slots to detect that $i+j \geq 2$ and $i \geq 2$, plus L_r-1 slots on average to resolve the collision (which is known in advance) in E'.

(3.3) is explained similarly.

By rearranging the terms in (3.2) and (3.3) we obtain that each of the equations is of the form:

$$x_N = \sum_{r=2}^{N}P(i=r)x_r + P(i=1)x_{N-1} + P(i=0)x_N + [P(i=0)+P(i=1)]\sum_{k=2}^{\infty}P(j=k)x_k + b_N \qquad N \geq 2 \tag{3.5}$$

where in (3.2):

$$b_N = 2 - P(i=0) + P(i=1)[\,1 - P(j=0) + P(j=1)\,] \triangleq b_N^{(L)} \tag{3.6a}$$

and in (3.3):

$$b_N = P(i=1) + P(j=1)[\,P(i=0) + P(i=1)\,] \triangleq b_N^{(M)} \tag{3.6b}$$

(3.5) defines an infinite system of equations in x_N that can be rewritten as:

$$x_N = \sum_{i=2}^{\infty}a_{N,i}x_i + b_N \triangleq F_N(X) \qquad N \geq 2 \tag{3.7}$$

where $X=\{x_N\}_{N=2}^{\infty}$ and $a_{N,i}$ and b_N are defined by (3.5) and (3.6).

Clearly, each of the sequences $\{L_N\}_{N=2}^{\infty}$ and $\{M_N\}_{N=2}^{\infty}$ is a solution of the respective system of equations (3.7). Employing a similar approach to that in [12], we state the following theorem:

Theorem: If

$$2q(1-q)>z(1-e^{-z}) \tag{3.8}$$

then:

a) The system of equations (3.7) has a solution $X'=\{x'_N\}_{N=2}^{\infty}$ which satisfies $0\le x'_N \le \alpha N \quad N\ge 2$ where $\alpha=4/[2q(1-q)-z(1-e^{-z})]$;

b) X' is unique in the class of sequences X that satisfy:

$$\lim_{k\to\infty} \max_{N<k} \sum_{i=k}^{\infty} a_{N,i} |x_i| = 0 \tag{3.9}$$

c) If for all $N\ge 2$ $b_N=b_N^{(L)}$ $(b_N=b_N^{(M)})$, then for all $N\ge 2$ $L_N=x'_N$ $(M_N=x'_N)$.

The Proof of this Theorem is given in [13].

It follows from the theorem that for all q and z satisfying (3.8), L_i and M_i do not increase with i more rapidly than linearly. On the other hand, for z not large, $a_{N,i}$ decrease very rapidly with increasing i for $i>N$ (like $i! \simeq i^i$). Hence, an approximate solution for L_N and M_N for $2\le N\le K$, where K is some integer, can be obtained by solving the finite system of equations obtained from the given infinite one by discarding all equations and unknowns commencing with $K+1$ (K is determined according to the desired accuracy).

L_N and M_N discussed above, correspond to a *CRI* that exactly N messages were transmitted in its first slot, that is, the new interval E_i (whose length is x) that corresponds to the *CRI*, contains exactly N messages. We denote the corresponding averages over N (which is Poisson distributed with parameter $W\triangleq\lambda x$) by:

$$L = \sum_{N=0}^{\infty} \frac{e^W W^N}{N!} L_N \quad ; \quad M = \sum_{N=0}^{\infty} \frac{e^W W^N}{N!} M_N \tag{3.10}$$

Let $M(s)$ denote the random variable which is the number of successfully transmitted messages during the first s *CRI*'s, and let $L(s)$ denote the random variable which is the number of slots required to transmit them. The throughput of the algorithm is defined as $T=\lim_{s\to\infty}\{E[M(s)/E[L(s)]]\}$. By using the fact that Poisson arrivals in disjoint intervals are independent, it is easy to show that $T=M/L$. Numerical optimization over q,z,W for achieving maximum throughput yields the following results: Throughput = 0.4892, for $W=1.24$, $q=0.26$, $z=0.66$. Note that the throughput achieved is higher than 0.4878. Note also that q and z do satisfy (3.8).

IV. AN ALGORITHM FOR THE KNOWN-MULTIPLICITY FEEDBACK

A. Operation of the Algorithm

The algorithm is the same as the *CRAI* algorithm in [11], except the way intervals of multiplicity 2 are treated. That is, the algorithm refers to two time axes - the arrival axis which shows the Poisson arrival instants, and the transmission axis which is segmented into slots. The system starts operating at time zero in both axes. In the first slot of the transmission axis, the interval $E_1=[0,x)$ of the arrival axis is enabled (x is a parameter to be optimized for achieving maximum throughput), and thereafter the algorithm initiates a certain process (that will be presently described), which terminates at some slot i. This operation is repeated for the interval $E_2=[x,2x)$ starting from the $(i+1)$-st slot, and so on.

The process initiated is as follows: If $[E_k]=0$ or 1, then the process terminates immediately. If $[E_k]=N\ge 2$, then the collision is resolved as in the *CRAI* algorithm (i.e., the interval is split into two subintervals according to a parameter q_N, and each of the subintervals is similarly resolved with the respective parameter), except the following: A memory cell, C, is used (at $t=0$ C is empty). Whenever an interval of multiplicity 2 is detected (which is contained in or equal to E_k), the process checks the state of C – if C is empty, the process inserts the interval to it, and continues as in the *CRAI* algorithm (without resolving the interval just inserted into C); if C is full, the process takes out the interval contained in it (and now C is empty), and resolves the two intervals together according to a resolution process that will be described later on, and thereafter continues as in the *CRAI* algorithm. The process terminates when all N messages contained in E_k have been successfully transmitted, except possibly a single interval of multiplicity 2, which is contained in C.

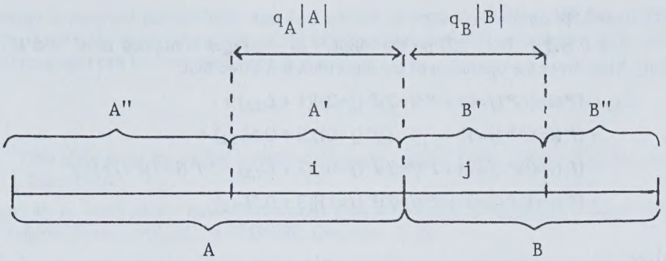
B. The Resolution Process

Before describing the resolution process of two intervals of multiplicity 2 formally, it will be helpful to give an informal description. Referring to Fig. 3(a), A and B are the intervals to be resolved. In the first step, the process enables the subintervals A' and B' together. All possible outcomes and their corresponding possible ways of distributing the messages among the subintervals A', B', A'', and B'', are listed in Fig. 3(b). If the outcome is $[A' \cup B']=0$ or 4, then the two empty subintervals are not treated any longer, and the process is repeated for the two subintervals of multiplicity 2. If the outcome is $[A' \cup B']=1$, then the set marker indicates which is the subinterval that contains the message (A' or B'), and therefore it is known immediately which is the other subinterval (A'' or B'') that contains a single message, and in the next step this subinterval is enabled. The remaining subinterval, which is of multiplicity 2, is resolved by the same process (when another such interval is detected). If the outcome is $[A' \cup B']=2$, then in the next step A'' is enabled – if $[A'']=0$ or 2, then the process is repeated for the two subintervals of multiplicity 2 that have been detected; if $[A'']=1$, then the three remaining subintervals, A', B', B'', each of which contains a single message, are enabled in turn. Finally, if the outcome is $[A' \cup B']=3$, then it is clear that $[A'' \cup B'']=1$, and in the next step this combination is enabled. Now, again, the set marker indicates which is the subinterval that contains the message (A'' or B''), and therefore it is known immediately which is the other subinterval (A' or B') that contains a single message, and in the next step this subinterval is enabled. The remaining subinterval is resolved by the same process.

We now give a formal description of the resolution process. The process uses a flag D that can be either ON or OFF, and is defined by the following sequence of steps.

The Resolution Process

```
(1) Check the state of $D$,
        If in OFF: Denote the interval that was taken out of $C$ by $A$, and the other interval by $B$;
                   switch $D$ to ON; goto (2)
        If in ON: Denote the interval that was taken out of $C$ by $B$, and the other interval by $A$;
                  switch $D$ to OFF; goto (2)
(2) Enable a right-hand subinterval $A'$ of $A$ whose length is $|A'|=q_A|A|$ $(0<q_A<1)$ together with a
    left-hand subinterval $B'$ of $B$ whose length is $|B'|=q_B|B|$ $(0<q_B<1)$,
        If $[A' \cup B']=0$: Denote $A''$ by $A$ and $B''$ by $B$ and goto (2)
        If $[A' \cup B']=1A'$: Enable $A''$; insert $B''$ to $C$; terminate
        If $[A' \cup B']=1B'$: Enable $B''$; insert $A''$ to $C$; terminate
        If $[A' \cup B']=2$: Goto (3)
        If $[A' \cup B']=3$: Goto (4)
        If $[A' \cup B']=4$: Denote $A'$ by $A$ and $B'$ by $B$ and goto (2)
(3) Enable $A''$,
        If $[A'']=0$: Denote $A'$ by $A$ and $B''$ by $B$ and goto (2)
        If $[A'']=1$: Enable $A', B', B''$, one after the other; terminate
        If $[A'']=2$: Denote $A''$ by $A$ and $B'$ by $B$ and goto (2)
(4) Enable $A''$ together with $B''$,
        If $[A'' \cup B'']=1A''$: Enable $A'$; insert $B'$ to $C$; terminate
        If $[A'' \cup B'']=1B''$: Enable $B'$; insert $A'$ to $C$; terminate
```



[A'UB']	[A''] = 2-i	[A'] = i	[B'] = j	[B''] = 2-j
0	2	0	0	2
1	1	1	0	2
	2	0	1	1
2	0	2	0	2
	1	1	1	1
	2	0	2	0
3	1	1	2	0
	0	2	1	1
4	0	2	2	0

Fig. 3: Mixing two intervals of multiplicity 2:
(a) Intervals partition
(b) Messages distributing alternatives

C. Performance Evaluation

Let $L_{2,2}$ denote the mean number of slots required to resolve two intervals of multiplicity 2. Let $\alpha L_{2,2}$ and $\beta L_{2,2}$ ($0<\alpha,\beta<1;\alpha+\beta=1$) be the contributions to $L_{2,2}$ of the left-hand interval and the right-hand interval, respectively (see Fig. 3(a)). As a consequence of step (1) in the resolution process, each interval of multiplicity 2 inserted to C at some step of the process, has equal probability to be a left-hand or a right-hand interval in its resolution process. Hence the mean number of slots required to resolve such interval is:

$0.5\alpha L_{2,2}+0.5\beta L_{2,2}=0.5L_{2,2}$.

Let i and j ($0 \le i,j \le 2$) be the number of messages contained in A' and B', respectively (see Fig. 3(a)). Then from the operation of the algorithm it follows that:

$$\begin{aligned} L_{2,2} = {} & [P(i=0)P(j=0) + P(i=2)P(j=2)](1 + L_{2,2}) + \\ & + [P(i=0)P(j=1) + P(i=1)P(j=0)](2 + 0.5L_{2,2}) + \\ & + [P(i=0)P(j=2) + P(i=2)P(j=0)](2 + L_{2,2}) + 5 \cdot P(i=1)P(j=1) + \\ & + [P(i=1)P(j=2) + P(i=2)P(j=1)](3 + 0.5L_{2,2}) \end{aligned} \tag{4.1}$$

where

$$P(i=r) = \binom{2}{r} q_A^r (1-q_A)^{2-r} \qquad r=0,1,2 \tag{4.2a}$$

$$P(j=s) = \binom{2}{s} q_B^s (1-q_B)^{2-s} \qquad s=0,1,2 \tag{4.2b}$$

The explanation of (4.1) is similar to that of (3.2).

Numerical minimization over q_A and q_B yields $L_{2,2}^{(min)}=5.580$ for $q_A^{(o)}=q_B^{(o)}=0.35$. The rest of the analysis is done by following a procedure parallel to that in [11], except that here $L_2=0.5L_{2,2}^{(min)}=2.790$ instead of $L_2=3$ as in [11]. This yields that the optimal choice is $x \to \infty$, and that for this choice, the algorithm is stable for $\lambda<0.5533$. Note that this is higher than 0.5324.

V. DISCUSSION AND SUMMARY

In this paper, we introduced a new scheme, in which extra information added to the message is exploited when the message is received successfully. We proposed a type of information to be added to the message, and a new algorithm for the ternary feedback model which exploits this information obtained during successful transmissions. This algorithm mixes a collision interval with a new interval, and achieves throughput of 0.4892, that is, higher than 0.4878 which is the highest throughput achieved so far for the same model.

Note that our scheme can be viewed as a channel coding scheme, in the sense that extra information added to the message is exploited to improve the performance of the system.

In the algorithm for the ternary feedback model, the bookkeeping concerning collision intervals and new intervals, requires an unbounded amount of memory. To overcome this difficulty, we first observe that the bookkeeping concerning new intervals differs substantially from that for collision intervals. The memory needed for the former, can be reduced to that needed for keeping a single point of the arrival axis, simply by appropriate shifting of clocks (without any degradation in the throughput). To bound the amount of memory required for the latter, one must limit the size of the *CIL* and resolve collision intervals that otherwise would have been appended to the *CIL*, by the 0.487-algorithm [4], whenever the *CIL* becomes full. This leads to a degradation in the throughput, being in the worst case (i.e. *CIL* of size zero) 0.4881, which is still higher than 0.4878.

We also proposed a new algorithm which exploits the same type of extra information obtained during successful transmissions, for the known-multiplicity feedback model. This algorithm mixes two intervals of multiplicity 2, and achieves throughput of 0.5533, that is, higher than 0.5324 which is the highest throughput achieved so far for this model. Mixing also two intervals of multiplicity 3 yields only a marginal additional increase in the throughput (an order of magnitude less than that achieved by mixing two intervals of multiplicity 2). It seems that it is not worthwhile to mix intervals of higher multiplicities.

Throughout the computations of the throughput, the set marker has not been separated from the message. Therefore, the results obtained are asymptotic with the message length. However, since the message length in the model is arbitrary, this doesn't arise any difficulty. Moreover, since the set marker is only a single bit, this doesn't cause any practical difficulty, too.

The improvements achieved in the throughput are not large, but note that the set marker and the algorithms proposed are only examples of a scheme in which extra information added to the message is exploited when the message is received successfully, and they are not necessarily the optimal choice. The importance of our results is not only in the magnitude of the improvement, but also and mainly in the fact that they prove that the throughput can be increased by such a scheme.

References

[1] J. I. Capetanakis, "Tree algorithms for packet broadcast channels", *IEEE Trans. Inform. Theory,* vol. IT-25, pp. 505-515, Sept. 1979.

[2] B. S. Tsybakov and V. A. Mikhailov, "Free synchronous packet access in a broadcast channel with feedback", *Prob. Inform. Trans.* , vol. 14, pp. 259-280, Oct.-Dec. 1978

[3] J. L. Massey, "Collision resolution algorithms and random-access communications", in *Multi-User Communications* (CISM Courses and Lectures Series), G. Longo, Ed., New York: Springer-Verlag, 1981.

[4] R. G. Gallager, "Conflict resolution in random access broadcast networks", in *Proc. AFOSR Workshop Communication Theory and Applications* , Provincetown, MA, Sept. 1978, pp. 74-76.

[5] B. S. Tsybakov and V. A. Mikhailov, "Random multiple access of packets: Part-and-try algorithm", *Prob. Inform. Trans.* , vol. 16, Oct.-Dec. 1980.

[6] J. Mosley and P. A. Humblet, "A class of efficient contention resolution algorithms for multiple access channels", *IEEE Trans. Commun.* , vol. COM-33, pp. 145-151, Feb. 1985.

[7] L. Merakos and D. Kazakos, "Multiaccess of a slotted channel using a control mini-slot",in *Proc. Int. Conf. Commun.* , Boston, MA, June 1983.

[8] J. C. Huang and B. Z. Xu, "Pretested random tree protocol for packet conflict resolution", *IEEE Trans. Inform. Theory,* vol. IT-29, pp. 910-914, Nov. 1983.

[9] J. C. Huang and T. Berger, "Delay analysis of interval-searching contention resolution algorithms", *IEEE Trans. Inform. Theory,* vol. IT-31, pp. 264-273, March 1985.

[10] S. S. Panwar, D. Towsley and J. K. Wolf, "On the throughput of degenerate intersection and first-come first-served collision resolution algorithms" *IEEE Trans. Inform. Theory,* vol. IT-31, pp. 274-279, March 1985.

[11] L. Georgiadis and P. Papantony-Kazakos, "A collision resolution protocol for random access channels with energy detectors", *IEEE Trans. Commun.* , vol. COM-30, pp. 2413-2420, Nov. 1982.

[12] B. S. Tsybakov and N. D. Vvedenskaya, "Random multiple-access stack algorithm", *Prob. Inform. Trans.* , vol. 16, pp. 230-243, July-Sept. 1980.

[13] I. Kessler and M. Sidi, "Mixing collision resolution algorithms exploiting information of successful messages", EE Pub. No. 602, Technion - Israel Institute of Technology, Haifa, Israel, Oct. 1986.

INSTABILITY CONDITIONS ARISING IN ANALYSIS OF SOME MULTIACCESS PROTOCOLS

WOJCIECH SZPANKOWSKI and VERNON REGO
Department of Computer Science Purdue University
West Lafayette, IN 47907 U.S.A.

Abstract. *An important question in stochastic modeling is whether a denumerable state Markov chain describing a system is stable or not. Adopting a general definition of stability we present some conditions for instability. We slightly extend previously obtained conditions for non-ergodicity, and we present criteria for moments of a chain to be infinite . Finally, we apply these instability conditions to study some multidimensional Markov chains describing multiaccess protocols in a broadcast environment. In particular, we show that finite-dimensional exponential back-off algorithm is nonergodic for any* $\lambda > 0$, *and the infinite-dimensional back-off algorithm is unstable for* $\lambda > 0.461$. *In addition, we prove that the decentralized dynamic control algorithm is unstable for* $\lambda > e^{-1} = 0.368$.

1. INTRODUCTION

The stability and instability of many important systems is usually determined by the classification of a related Markov chain. The question of ergodicity, recurrence and transience of a Markov chain is of great importance; however, in practice a more general definition of stability is necessary. It is said that a system is stable if it possesses *required properties* in the presence of some *disturbances*. For stochastic models disturbances are usually associated with the input process, and stability or instability depend upon the definition of the required property. For example, assuming existence of steady-state distribution we deal with the ergodicity of a Markov chain; by considering finiteness of some moments we investigate a new type of stability. This paper discusses the instability property of a Markov chain, and we focus our attention on nonergodicity and infiniteness of some moments of the process.

Let N^t $(t = 0,1, \ldots,)$ denote an irreducible aperiodic Markov chain with state space C over the nonnegative integers. Let $P = (p_{ij})_{i,j \in C}$ be its transition matrix and π_i, $i \in C$ the steady-state probabilities of the chain. It is well known that for ergodic Markov chains the steady state probabilities are positive for all $i \in C$ and they satisfy the following system of linear equations

$$\pi_j = \sum_{i \in C} \pi_i \, p_{ij}, \quad j \in C. \tag{1}$$

A simple criteria for ergodicity are available through the so called Lyapunov (or test) function approach (see for example Foster [1953], Tweedie [1976]). Let $V(k)$, $k \in C$ be a nonnegative real-valued function called a Lyapunov function, and we denote by $AV(k)$, $k \in C$ an operator for a Markov chain with respect to that function if it exists. That is,

$$AV(k) = E\{V(N^{t+1}) - V(N^t) \,|\, N^t = k\}, \quad k \in C. \tag{2}$$

The operator $AV(k)$ is often called a generalized drift, since for $V(k) = k$ it becomes the average drift $d(k) = E\{N^{t+1} - N^t \,|\, N^t = k\}$. If the generalized drift is negative for all but finitely many states, then the Markov chain is ergodic (Foster [1953], Tweedie [1976]). Kaplan [1979], Sennott et al [1983] and Szpankowski [1985] proved that, under some simple condition on the test function, $AV(k) \geq 0$ for all but finitely many states implies nonergodicity , if an additional condition known as the (generalized) Kaplan condition is satisfied. However, in practice it is sometimes difficult to verify the (generalized) Kaplan condition (e.g. for multidimensional Markov chains).

In the next section, we show that under the same condition as in Sennott et al [1983] and Szpankowski [1985], the criterion $AV(k) \geq 0$ for infinitely many states (i.e., the condition may be violated for an infinite number of states) together with Kaplan condition is sufficient for nonergodicity. Moreover, if we drop the Kaplan condition in the last statement, then we shall prove that such a modified criterion implies $\lim_{t \to \infty} EV(N^t) = \infty$. These criteria find useful applications in instability analysis of some multidimensional Markov chains. In particular, we apply them to study instability problems of multiaccess protocols, e.g., exponential back-off algorithms (Rosenkrantz [1984], Kelly [1985]) and a decentralized dynamic control algorithm (Hajek and van Loon [1982]).

2. MAIN RESULTS

Criteria for nonergodicity involves the following function (see Sennott et al [1983] and Szpankowski [1985]) :

$$\psi_k^V(z) = (z^{V(k)} - \sum_{j \in C} p_{kj}\, z^{V(j)})/(1-z), \quad z \in [0,1), \quad k \in C. \tag{3}$$

Note that for $|AV(k)| < \infty$ by *l'*Hospital rule $\lim_{z \to 1} \psi_k^V(z) = AV(k)$. The next theorem generalizes the results form Sennott et al [1983] and Szpankowski [1985].

THEOREM 1. Let N^t be an irreducible Markov chain, and $V(k)$ a Lyapunov function such that for a proper subset H of C the following holds

$$\inf_{k \in C-H} V(k) > \sup_{k \in H} V(k), \tag{4}$$

Let also $H_1 \subset C$ be a finite subset of C. If $|AV(k)| < \infty, k \in C$ and

$$AV(k) \geq 0 \quad k \in C-H \tag{5}$$

$$\psi_k^V(z) \geq -B \quad k \in C-H_1, \tag{6}$$

where $B \geq 0$ is a constant, then the chain is not ergodic.

Proof. Assume the contrary that N^t is ergodic and let $\pi_k,\ k \in C$ be its steady state distribution. Then, by (3) we find that $\sum_{k \in C} AV(k)\, \psi_k^V(z) = 0$. But, by (6) and Fatou's Lemma

$$0 = \lim_{z \to 1} \sum_{k \in C} \pi_k\, \psi_k^V(z) \geq \sum_{k \in C} AV(k)\pi_k = \sum_{k \in H} AV(k)\pi_k + \sum_{k \in C-H} AV(k)\pi_k > 0.$$

The last inequality follows from (5) and a generalization of Proposition 1 in Sennott et al [1983] (see

Szpankowski and Rego [1986]). This is the desired contradiction.

□

Remarks:

(i) In Theorem 1 H might be any infinite subset of C satisfying (4). This generalizes previous results, where H was assumed to be finite.

(ii) If in (4) we replace strict inequality by a weak one, then for validity of Theorem 1 we must have strict inequality in (5) for some $k \in C - H$.

(iii) Another extension of Theorem 1 is possible if one notes that introducing a parametric Lyapunov function $V_k(z) = z^{V(k)}$, $k \in C$, $z \in [0,1)$ and a function $g(z) = 1 - z$, we may represent (3) as $\psi_k^V(z) = -AV_k(z)/g(x)$, and (5) is equivalent to $-AV'_k(1)/g'(1) \geq 0$ where $AV'_k(1)$ is the operator of the derivative of $V(z)$ at $z_0=1$. This approach was adopted in Szpankowski [1985], where a precise definition of a generalized parametric Lyapunov function $V_k(z)$ and additional function $g(z)$ were introduced , and conditions (4), (5) and (6) are expressed in terms of $AV_k(z)$ as above. These functions must satisfy some conditions, a few of which are that $V_k(z)$ is uniformly bounded for all $k \in C$, $z \in [a,b]$, and there exists such $z_0 \in [a,b]$ that $V_k(z_0) = \text{const}$, and $g(z_0) = 0$ (for details see Szpankowski [1985]). A particular application of this is as follows. Let us define a new function $L_k(z) = \sum_{m=0}^{n} n_{[n-m]} V_k^{(m)}(z)(1-z)^m$, $g(z) = (1-z)^{n+1}$ where n is an integer, $n_{[n-m]} \overset{\text{def}}{=} n(n-1) \cdots (n-m+1)$, and $V_k^{(m)}(z)$ is the m-th derivative of $V_k(z)$ with respect to z. Then, $-AL'_k(z) = -AV_k^{(n+1)}(z)(1-z)^n$, $g'(z) = (n+1)(1-z)^n$ and applying Proposition 1 of Szpankowski [1985] with $z_0 = 1$ one finds that a Markov chain N^t is not ergodic if $-AV_k^{(n+1)}(1) \geq 0$ for $k \in C - H$ and $-AL_k(z) \geq -B(1-z)^{n+1}$, $k \in C - H_1$, $B \geq 0$ where the subsets H and H_1 are defined above in Theorem 1. In particular, if $V_k(z) = z^k$ then the first condition is equivalent to $\sum_{j \in C} p_{kj} \{ j(j-1) \cdots (j-n) - k(k-1) \cdots (k-n) \} \geq 0$ for $k \geq M$, where M is a constant, and it is proved (Kaplan [1983], Sennott [1985]) that the Kaplan condition is reduced to $k^n \psi_k(z) \geq -B$ where $\psi_k(z)$ is given by (3) with $V(k) = k$.

□

The applicability of Theorem 1 depends on verification of conditions (4), (5) and (6). In most cases (4) and (5) are easy to check, but (6) needs some additional work. However, it is proved (Kaplan [1979], Sennott et al [1983], Szpankowski [1985]) that for downward uniformly bounded Markov chains ($p_{kj} = 0$ for $j < k - m$, m is given) condition (6) is automatically satisfied. Nevertheless, in the general case a question arises: what are properties of the Markov chain that follow from conditions (4) and (5) alone?

We now ask under what conditions $\lim_{t \to \infty} E\, V(N^t) = \infty$. In other words, we present criteria that assure instability of a Markov chain in the sense that the average value of N^t over a function $V(k)$ is infinite for t tending to infinity. In particular, if $V(k) = k^r$, r is a real number, then the criteria establish conditions under which the r-th moment of N^t in a steady state is unbounded. The proofs are omitted and the reader may consult Szpankowski and Rego [1986].

We start with an ergodic case, that is, we first assume that N^t is ergodic, and establish conditions under which moments of N^t are unbounded.

THEOREM 2. Let N^t be an irreducible, *ergodic* Markov chain, and $V(k)$ a nonnegative Lyapunov function such that

$$\inf_{k \in C-H} V(k) > \sup_{k \in H} V(k), \tag{7}$$

for some $H \subset C$. If

$$AV(k) \geq 0 \quad k \in C-H, \tag{8}$$

then

$$\lim_{t \to \infty} EV(N^t) = \infty. \tag{9}$$

□

We prove now that for nonergodic Markov chains (7) and a slightly modified (8) imply (9). It easy to show that (7) and $A\,V(k) \geq \varepsilon > 0$ imply that $V(k)$ is unbounded. Hence, a simple analysis reveals that

THEOREM 3. Let N^t be an irreducible *nonergodic* Markov chain satisfying (7), and for a subset $H \subset C$ the following holds

$$A\,V(k) \geq \varepsilon > 0 \quad k \in C-H \tag{10}$$

where ε is a small positive number. If, in addition,

$$\lim_{j \to \infty} V(j) = \infty \tag{11}$$

then (9) holds. If instead of (11) we assume

$$\sum_{j \in C} p_{j,k} V(j) < \infty \tag{12}$$

then (7), (10) and (12) imply that

$$\limsup_{t \to \infty} E\,V(N^t) = \infty \tag{13}$$

□

Finally, Theorem 2 and 3 imply our main result

PROPOSITION. For an irreducible Markov chain satisfying (7), (10) and (11) condition (9) holds. If we replace (11) by (12), then (13) is satisfied.

□

Remarks

Conditions of the above Proposition are sufficient but not necessary. We can find a Markov chain with negative drift for all but one state such that the chain does not possess any moment. To show this , assume that $p_{nm} = 0$ for $m \neq 0$, $n+1$ and $p_{n,n+1} = p_n$, $p_{n,0} = q_n = 1 - p_n$ where $p_n < n/(n+1)$ for $n \neq 0$. Then $d(n) = (n+1)p_n - n < 0$ for $n \neq 0$, hence by Foster's criterion the chain is ergodic. Solving (1) for $p_n = (n-1)/(n+1)$ for $n \geq 2$ and $p_0 = p_1 = 0.5$ we show that $\pi_n = 0.5\ \pi_0/[n(n-1)]$, $n \neq 0, 1$, $\pi_0 > 0$. Then for any $r \geq 1$

$$EN^r \leq \frac{\pi_0}{2} \sum_{n=2}^{\infty} \frac{n^{r-1}}{n-1} = \infty,$$

hence the chain does not possess any moments EN^r, $r \geq 1$.

□

3. APPLICATIONS TO INSTABILITY OF SOME MULTIACCESS PROTOCOLS

Assume that an infinite number of users sharing a common communication channel transmit fixed-length packets. The channel is slotted and a slot duration is equal to a packet transmission time. When two or more packets transmit at the same time, then there is a collision and no packet is transmitted. There is a variety of protocols for resolving these collisions, and they differ depending on how feedback information for the channel is used to resolve the collisions. To avoid subsequent collisions a probability of retransmitting a collided packet is introduced, which controls the number of retransmissions. This probability depends on the outcomes of the channel, the time, the number of users involved in a collision, etc. We use a letter f as a generic notation for this probability with an index to show a relationship between the probability and the state of the system. We describe below some examples of multiaccess protocols for which we investigate instability conditions.

EXAMPLE 1. *Finite-dimensional back-off (FDB) algorithm*
Kelly [1985], Hajek [1982]

We assume that each user has a counter which contains the number of times a packet was involved in a collision. Let f_k, $k = 0, 1, \ldots, B$ be the probability of transmitting a packet where k is the number of times a packet has already been unsuccessfully transmitted. Hajek [1982a] assumed that B is finite and after B unsuccessful transmissions a packet is rejected and declared lost (so called *acknowledgement based retransmission control* algorithm). In our algorithm, we assume that for the FDB algorithm, after B unsuccessful transmissions a packet is retransmitted with a constant probability f_B until success occurs. By this assumption, we can uniformly model the original ALOHA protocol for which $f_0 = 1, f_1 = r, B = 1$, and the ALOHA protocol without retransmission discrimination policy, where $f_0 = r, B = 0$ (Szpankowski [1984]). For the finite-dimensional exponential back-off algorithm we assume $f_k = 2^{-k}, 0 \leq k \leq B < \infty$.

EXAMPLE 2. *Infinite-dimensional back-off algorithm* (IDB)
Rosenkrantz [1984], Kelly [1985], Goodman et al [1985]

This is exactly the same algorithm as in Example 1, except that $B = \infty$. Exponential back-off is characterized by $f_k = 2^{-k}$, $0 \leq k \leq \infty$, and for linear back-off we assume $f_0 = 1$, $f_k = 1/k$, $1 \leq k < \infty$.

EXAMPLE 3. *Decentralized dynamic control algorithm*
Hajek and van Loon [1982], Kelly [1985]

Another strategy to select the probabilities f was proposed by Hajek and van Loon [1982]. Now each user contains a counter, S_t, $t = 0,1, \ldots,$ which is updated recursively at the end of each slot according to some rules common for all users. For example, Hajek and van Loon [1982] assumed $S_{t+1} = \max\{S_t, 1, a(Z_t)\}$ where Z_t is an outcome from the channel (idle, success or collision) and $a(\cdot)$ is a function of Z_t. On the other hand, Kelly [1985] proposed $S_{t+1} = \max\{1, S_t + a\ I[Z_t = 0] + B\ I[Z_t = 1] + c\ I[Z_t = \text{collision}]\}$, where a, b, c are constants and $I(\cdot)$ is an indicator function of an event. The probability of transmitting a packet is the same for all users at a time t and is equal to $f_t = 1/S_t$.

□

We present below some instability analyses for the above algorithms.

Finite-dimensional back-off algorithm

In this case the system is described by a B-dimensional Markov chain $\mathbf{N}^t = (N_1^t, N_2^t, \ldots, N_B^t)$ where N_k^t is the number of packets at the beginning of time t that have been unsuccessfully transmitted k times. Let $A(t)$ denote the number of new packets that arrive in a slot t. Assume $A(t)$, $t = 0,1, \ldots,$ is i.i.d. process and $c_i = Pr\{A(t)=i\}$, $\lambda = EA(t)$. Under this assumption $\mathbf{N}^t$ is Markov chain and the question of N^t's ergodicity arises.

To show instabilty conditions for the system, let $\mathbf{n} = (n_1, n_2, \ldots, n_B)$ and we introduce the Lyapunov function $V(\mathbf{n}) = \sum_{i=1}^{B} n_i$. Note that $AV(\mathbf{n}) = \sum_{i=1}^{B} d_i(\mathbf{n})$ where $d_i(\mathbf{n}) = E\{N_i^{t+1} - N_i^t \mid \mathbf{N}^t = \mathbf{n}\}$, is the i-th component of the average drift-vector of $\mathbf{N}^t$. Denoting

$$f_\pi(\mathbf{n}) \overset{\text{def}}{=} \prod_{l=1}^{B} (1 - f_l)^{n_l}, \tag{14}$$

we immediately obtain (see also Rosenkrantz [1984]),

$$d_1(\mathbf{n}) = \lambda - f_1 n_1 - c_1 f_0 f_\pi(\mathbf{n}), \tag{15a}$$

$$d_k(\mathbf{n}) = f_{k-1} n_{k-1} - f_k n_k - c_0 n_{k-1} f_{k-1}(1 - f_{k-1})^{-1} f_\pi(\mathbf{n}), \quad 2 \le k \le B-1 \tag{15b}$$

$$d_B(\mathbf{n}) = f_{B-1} n_{B-1} - c_0 f_\pi(\mathbf{n}) \left[n_{B-1} f_{B-1}(1 - f_{B-1})^{-1} + n_B f_B (1 - f_B)^{-1} \right], \tag{15c}$$

and

$$AV(\mathbf{n}) = \lambda - f_\pi(\mathbf{n}) \left[c_1 f_0 + c_0 \sum_{k=1}^{B} n_k \frac{f_k}{1 - f_k} \right], \tag{16}$$

which states that the average drift is equal to the input rate minus average throughput.

To study instability of the algorithm we must investigate properties of the following expression $f_\pi(\mathbf{n}) \sum_{k=1}^{B} n_k f_k/(1 - f_k)$, with $f_\pi(\mathbf{n}) = \prod_{l=1}^{B} (1 - f_l)^{n_l} = \exp\left[- \sum_{l=1}^{B} n_l \ln(1 - f_l)^{-1} \right]$. Let us introduce a function

$$g(\mathbf{x}) = \left[\sum_{l=1}^{B} a_i \; x_i \right] \exp\left[- \sum_{l=1}^{B} b_i \; x_i \right], \tag{17}$$

where $a_i, b_i > 0$, $i = 1, \dots, B$ are nonnegative constants and $\mathbf{x} = (x_1, x_2, \dots, x_B) \geq 0$. In our case, $a_i = f_i/(1 - f_i)$, $b_i = \ln(1 - f_i)^{-1}$ and $x_i = n_i$, $i = 1,2, \dots, B$. We can prove (Szpankowski and Rego [1986]) that under the above constraints on **a**, **b**, and **x**

$$g(\mathbf{x}) \leq e^{-1} \max\{ \frac{a_1}{b_1}, \frac{a_2}{b_2}, \dots, \frac{a_B}{b_B} \}, \tag{18}$$

and the maximum value of $g(\mathbf{x})$ is reached at one of the boundary points $(b_1^{-1}, 0, \dots, 0)$ or $(0, b_2^{-1}, \dots, 0)$ or ... or $(0, 0, \dots, b_B^{-1})$. Let us also divide the Euclidean space $\mathbf{E} = \{\mathbf{x} : x_i \geq 0 \;\; i = 1,2, \dots, B\}$ into two subspaces $\mathbf{E}_1 = \{\mathbf{x} \in E : b_1 x_1 + \cdots + b_B x_B \leq 1\}$ and $\mathbf{E}_2 = \{\mathbf{x} \in E : b_1 x_1 + b_2 x_2 + \cdots + b_B x_B > 1\}$. Then, it is proved that $g(\mathbf{x})$ is 'decreasing' in E_2 in the sense that $g(\mathbf{x})$ defined on a line $L = \{\mathbf{x} \in E : x_1 = x_1, x_2 = \beta_2 x_1, \dots, x_B = \beta_B x_1, \beta_i \geq 0 \;\; i = 2,\dots,B\}$ is decreasing for $x \in L \cap E_2$. In particular, defining $H_M = \{x \in E : x_1 + x_2 + \cdots + x_M < M\}$ where $M = \max_{1 \leq i \leq B} \{b_i^{-1}\}$ we prove that for any $\varepsilon > 0$ there exists such an $M > 0$ that

$$g(x) < \varepsilon \;\; \text{for} \;\; x \in E - H_M. \tag{19}$$

Note also that for a given δ we can find such M_1 that for $\mathbf{n} \in C - H_{M_1}$ the following holds $f_\pi(\mathbf{n}) < \delta$. This and (19) imply (for details see Szpankowski and Rego [1986])

COROLLARY 1. For any finite B and $f_i > 0$ for all $i = 1,2, \dots, B$ the system is not ergodic for $\lambda > 0$.

□

In spite of the fact that the chain is not ergodic we can prove that $\lim_{t \to \infty} \sup EN_k^t \leq \lambda f_k^{-1} < \infty$ for $k = 1,2, \dots, B - 1$, and hence by the corollary, we must have $\lim_{t \to \infty} EN_B^t = \infty$.

Infinite-dimensional back-off algorithm

We assume now that $B = \infty$. This implies that N^t is an infinite dimensional Markov chain. Formulas (15a) and (15b) for the k-th component of the average drift are valid (however we do not need formula (15c)). In particular, (16) becomes

$$AV(\mathbf{n}) = \lambda - f_\pi(\mathbf{n}) \left[c_1 f_0 + c_0 \sum_{k=1}^{\infty} n_k \frac{f_k}{1 - f_k} \right], \tag{20}$$

and $f_\pi(\mathbf{n}) = \prod_{l=1}^{\infty} (1 - f_l)^{n_l}$. In this case the chain is obviously not downward uniformly bounded (it has infinite downward jumps), and therefore condition (6) in Theorem 1 should also be also verified.

Moreover, the set H_M defined above is infinite, and also $\lim_{k \to \infty} f_k = 0$. To avoid these difficulties (in particular, to bypass verification of Kaplan's condition (6)) we apply Proposition. We prove that

COROLLARY 2. Let $f_{\max} = \max_{1 \le l \le \infty} f_l, f_0 = 1$ and $\gamma = -\frac{1 - f_{\max}}{f_{\max}} \, ln(1 - f_{\max})$. If λ_0 is a solution of the following equation

$$\lambda - \frac{c_0}{\gamma} \exp [\gamma c_1/c_0) - 1] = 0, \qquad (21)$$

then for $\lambda > \lambda_0$ the average backlog grows linearly in time, that is, there exists such $\varepsilon > 0$ that $E \sum_{i=1}^{\infty} N_i^t \ge \varepsilon t$, assuming $N^0 = 0$.

Proof: We apply Proposition with H being an empty set, and $V(n) = \sum_{i=1}^{\infty} n_i$. Then $AV(n)$ is given by (16) with $B = \infty$. We can prove that

$$f_{\pi}(n) \le \exp \left[-\gamma \sum_{l=1}^{\infty} n_l \frac{f_l}{1 - f_l} \right] \qquad (22)$$

Let $s = \sum_{l=1}^{\infty} n_l \frac{f_l}{1 - f_l}$. Then by (22) and (16) we find

$$AV(n) = \lambda - f_{\pi}(n) \left[c_1 + c_0 \sum_{l=1}^{\infty} n_l \frac{f_l}{1 - f_l} \right] \ge \lambda - \max_{0 \le s < \infty} \{ e^{-\gamma s}(c_1 + c_0 s) \}. \qquad (23)$$

But the second term of (23) is equal to $\frac{c_0}{\gamma} \exp [\gamma c_1/c_0 - 1]$ for $s^* = 1/\gamma - c_1/c_0$. Hence, equation (21) follows. In fact, since H is empty we can prove that the total backlog grows linearly in time.

☐

In particular, for Poisson input traffic with $c_0 = e^{-\lambda}$, $c_1 = \lambda e^{-\lambda}$ and $f_{\max} = 0.5$ (e.g. binary exponential back-off algorithm and linear back-off) equation (21) reduces to

$$\lambda \ln 2 - \exp [\lambda (ln 2 - 1) - 1] = 0 \qquad (33)$$

and $\lambda_0 = 0.461$. For binary exponential back-off algorithm this result was also obtained by Sennot [1985a]. This can be compared with $\lambda_0 = 0.72$ for the same type of stability proved by Rosenkrantz [1984]. Note also that smaller the $f_{\max}$ is, smaller the λ_0 is. For example, for $f_l = 10^{-l}$ by equation (21) we find that $\lambda_0 = 0.381$. Finally, we can prove, using the same arguments as for finite-dimensional back-off algorithm, that $\lim_{t \to \infty} \sup EN_k^t \le \lambda f_k^{-1} < \infty$ (see Rosenbrantz [1984]).

Remarks.

(i) Kelly [1985] showed that any back-off algorithm with $f_0 = 1$ is not ergodic for $\lambda < 0.567$. (In his elegant and very simple proof he used only equilibrium arguments equating arrival and departure rates). In addition, he proved (see also Kelly and MacPhee [1986]) that for exponential back-off protocol the number of successful transmissions is infinite for $\lambda < \log 2 = 0.693$, and finite for $\lambda \ge 0.693$.

(ii) *Conjecture* *. Extension of Corrolary 1 to the case $B = \infty$ and $\inf_{1 \le k \le \infty} f_k = 0$ fails mainly because we cannot properly define the set H_M (M is infinite in that case) such that condition (7) is satisfied. However, we know that properties (18) and (19) of the function (17) (which plays an essential role in our proofs) still hold for the set $E_2 = \{x \in E : \sum_{k=1}^{\infty} b_i x_i > 1\}$ where $b_i = \ln(1 - f_i)^{-1}$. Therefore, we conjecture that Corollary 1 holds also for $B = \infty$ and $\inf_{1 \le k < \infty} f_k = 0$.

□

Decentralized dynamic control algorithm

As explained in Example 3, the system is described by a two dimensional Markov chain (N^t, S^t) where N^t is the backlog and S_t is a counter maintained by a user according to some recursive formula. The probability of transmitting a packet is the same for all blocked users and equal to $f_t = 1/S_t, S_t \ge 1$. The N-th component of the average drift is given by (see Kelly [1985])

$$d(n, s) = E\{N^{t+1} - N^t \,|\, N^t = n \;\; S^t = s\} = \lambda - \frac{n}{s}\left(1 - \frac{1}{s}\right)^{n-1}.$$

Applying Proposition with $H_M = \{(n,s) : n < M\}$ (note that H_M is infinite) one can easily prove (for details see Szpankowski and Rego [1986])

COROLLARY 3. If $\lambda > e^{-1}$, then for any recursive formula on S_t, $S_t \ge 1$, the average backlog is infinite, that is, $\lim_{t \to \infty} EN^t = \infty$.

□

Hajek [1982b] proved that for a special multiplicative recursion on S_t (see also Example 3 and Kelly [1985]), the Markov chain (N^t, S^t) is geometrically ergodic for $\lambda < e^{-1}$, and all moments of N^t in the steady state exist and are finite. We have shown above that for $\lambda > e^{-1} + \varepsilon$ the average backlog is infinite for any recursive formula on S^t.

REFERENCES

ALDOUS, D. 1986. Ultimate instability of exponential back-off protocol for acknowledgement-based transmission control of random access communication channels. preprint, submitted to *IIEEE Trans. on Information Theory.*

FOSTER, F. G. 1953. On the Stochastic Matrices Associated with Certain Queueing Processes. *Ann. Math. Stat.* **24** , 355 - 360.

GOODMAN, J., GREENBERG, A.G., MADRAS, N., MARCH, P. 1985. On Stability of the Ethernet. *Proc. of 1985 ACM Symposium on Theory of Computing, 379 - 387.*

---

* Recently, Aldous [1986] has proved that the infinite dimensional binary back-off algorithm is unstable for all $\lambda > 0$.

HAJEK, B. 1982a. Acknowledgement Based Random Access Transmission Control: An Equilibrium Analysis. *IEEE International Conference on Communications,* Philadelphia.

HAJEK, B. 1982b. Hitting-time and Occupation-time Bounds Implied by Drift Analysis with Applications. *Adv. Appl. Probab.,* **14** , 502 - 525.

HAJEK, B., van LOON, T. 1982. Decentralized Dynamic Control of a Multiaccess Broadcast Channel. *IEEE Trans. Automatic Control,* **AC27** , 559 - 569.

KAPLAN, M. 1979. Sufficient Condition for Nonergodicity of a Markov Chain. *IEEE Trans. Inform. Theory,* **IT-25** , 470 - 471.

KAPLAN, M. 1983. Non-ergodicity of Markov Chains. Another Look. INRS Report 1985.

KELLY, F.P. 1985. Stochastic Models of Computer Communication Systems. *J, R. Statistic. Soc.,* **47** .

KELLY, F.P., MACPHEE, I.M. 1986. The Number of Packets Transmitted by Collision Detect Random Access Schemes. *Submitted.*

ROSENKRANTZ, W.A. 1984. Some Theorems on the Instability of the Exponential Back-off Protocol. *Proc. of PERFORMANCE'84,* Paris France, 199 - 205.

SENNOTT, L.I., HUMBLET, P.A., TWEEDIE, R.L. 1983. Mean Drifts and the Non-ergodicity of Markov Chains. *Oper. Res.,* **31** , 783 - 787.

SENNOTT, L.I. 1985. Conditions for the Non-ergodicity of Markov Chains with Application to the Multi-Access Communication Systems. *Submitted.*

SENNOTT, L.I. 1985a. Techniques for Assessing the Stability of Multiple-Access Protocols. *23-rd Annual Allerton Conference on Communication, Control and Computing* University of Illinois at Urbana-Champain, 1985.

SZPANKOWSKI, W. 1984. Ergodicity Aspects of Multidimensional Markov Chains with Applications to Computer System Analysis. in: *Modelling and Performance Evaluation Methodology,* ed. F. Bacelli and G. Fayolle, Springer-Verlag, Berlin, 279-319.

SZPANKOWSKI, W. 1985. Some Sufficient Conditions for Non-ergodicity of Markov Chains. *J. Appl. Prob.,* **22** , 138 - 147.

SZPANKOWSKI, W., REGO, V. 1986. Some theorems on instability with applications to multiaccess protocols. submitted to a journal.

TWEEDIE, R.L. 1976. Criteria for Classifying General Markov Chains. *Adv. Appl. Prob.,* **8** , 737 - 771.

ANALYSIS OF ACCESS PROTOCOLS FOR A CENTRALIZED NETWORK

SANJAY UPPAL and JAMES A. BUCKLEW
Electrical and Computer Engineering Department
University of Wisconsin-Madison
Madison, Wisconsin 53706

ABSTRACT

We consider a communications network in which a large number of terminals send data to a centralized host. Such systems arise when electrical or water meters call in on phone lines to the host computer at the utility company. We propose and analyze three protocols that the meters use to access the phone.

DEVELOPMENT

In existing schemes, electrical and water meters are read periodically by persons sent by the utility company. Recently a system has been proposed in which the meters send the reading to the utility company over phone lines. An abstraction of the system is shown in the figure(Fig.1). The action taken by a meter when it has a message to send is as follows:

```
While message not transmitted do
  Sense the phone;
  If the phone is not in use then
        Make a call attempt;
        If the line is not engaged then
          Transmit the message;
        Else
          Delay attempt;
        End;
  Else
        Delay attempt;
  End;
End.
```

The problem consists of designing a suitable protocol that the meters use to access the phone and the switch.

Some notation used
A SLOT is one message duration. An EPOCH is the mean time between generation of messages at a meter. The LOADING FACTOR ρ is the fraction of the slots assigned over one epoch. For the random scheme it is S- the mean number of messages generated per slot by all the meters. A PERIOD is the number of slots in the call back duration.

The Fixed Scheme
In this scheme, all the slots for l periods are assigned followed by one or more empty periods(for the blocked callers to call back). A meter that is blocked at the switch waits a deterministic amount of time and attempts again. Thus this scheme consists of a contiguous collection of assigned slots, followed by another such collection of empty slots.

The Coin Flip Scheme
Meters are alloted to the slots by flipping a coin with a success probability of the loading factor. A blocked caller waits a deterministic amount of time before attempting again. This scheme results in the slots being uniformly assigned to the meters.

The Random Scheme
A meter call in whenever it has a ready message. Slots are not assigned to the callers. Since slots are not assigned, a meter may be blocked at the switch even there is no blocking at the phone.

Evaluation

Ease of Implementation: The Random scheme is the easiest to implement, as no assigning of slots is done. In both the Fixed and Coin Flip schemes, the meters have explicitly to be told when to call in.

Addition of new meters: An addition of a new caller to the fixed scheme disrupts the contiguous pattern of the slot assignment. For the Coin Flip scheme adding a meter is easier, since any unassigned slot will do. For the random scheme, nothing additional need be done for new meters as slots are not assigned.

Call attempts per successful call: This factor should be a minimum as each call attempt uses power from the limited source at the meters' disposal. An analysis was done to evaluate this factor assuming that the caller was blocked at the phone in its assigned slot. The random scheme behaved similar to the Coin Flip one at low values of S(a infinite population model was used). However, at higher values of S longer retransmission delays were required to approach the asymptotic limit shown below [1].

Fixed: $1/(2*(1-\rho))$ Coin Flip: $1/(1-\rho)$ Random: $1/(1-S)$

Extra calls generated by a host breakdown: If the host line were continually engaged, there would be a large glut of callers at the switch. Further, each attempt requires power as mentioned above. Therefore there is a need to minimize this quantity. In this analysis, a finite population model was assumed for the random scheme. The breakdown occurs for rx slots where x is also the deterministic retransmission delay for the first two schemes.

Fixed: $xr^2\rho/(2*(1-\rho))$ Coin Flip: $xr(r+1)\rho/(2*(1-\rho))$ Random: Same as Coin Flip for low $S \leq 0.3$

CONCLUSION

The order that the fixed scheme imposes enables a blocked caller to find an empty slot in the least time. For low values of the loading factor, the random scheme behaves similar to the coin flip one. For these values, it would be the preferred scheme due to the ease of implementation. Further, if the phone blocking is high, then any order imposed by the fixed or coin flip scheme is lost as most of the meters miss their assigned slots. However, if minimizing the number of call attempts per succesful call is critical then the fixed scheme is preferred.

REFERENCES

[1] L.Kleinrock and F.A. Tobagi, "Packet switching in radio channels: Part I-Carrier sense multiple access modes and their throughput-delay characteristics," IEEE Trans. Comm., vol. COM-23, pp.1400-1416, Dec.1975.

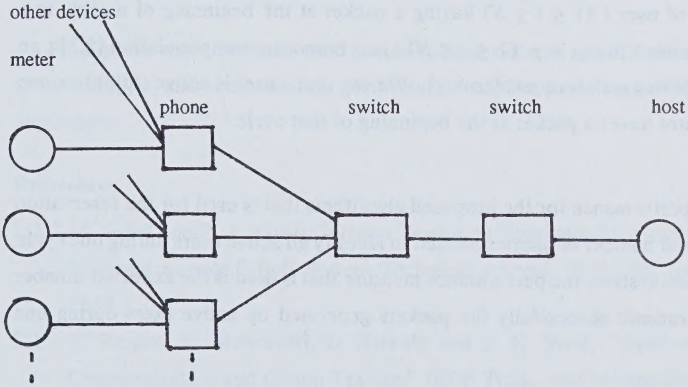


Fig.1

MULTIPLE ACCESS ALGORITHMS VIA GROUP TESTING FOR HETEROGENEOUS POPULATION OF USERS

DORON KURTZ and MOSHE SIDI[1]

Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa 32000, Israel

SUMMARY

In this paper we study the application of group testing techniques [1] to the design of efficient algorithms for multiple-access systems with *heterogeneous* population of users.

We assume that a population of N users use a slotted-time, collision-type common channel for transmission of information, and we consider two types of multiple-access systems: direct transmission systems and reservation systems. We assume that at the end of each slot the users are informed about what happened in that slot and distinguish between two types of feedback channels: binary feedback (Something/Nothing) and ternary feedback (idle, success or a collision slot).

The multiple-access algorithms that we propose operate cyclically; the collisions, if any, between packets that arrive during one cycle are resolved during the next cycle. We assume that the probability of user i $(1 \leq i \leq N)$ having a packet at the beginning of a cycle is p_i (previous studies assumed that $p_i = p$ $(1 \leq i \leq N)$, i.e. homogeneous population [2,3], an assumption that simplifies matters considerably). We say that a user is active (idle) in some cycle if it has (does not have) a packet at the beginning of that cycle.

The measure of performance for the proposed algorithms that is used for the reservation systems is the expected number of queries needed to identify all active users during one cycle. For direct transmission systems the performance measure that is used is the expected number of slots needed to transmit successfully the packets generated by active users during one cycle.

---

[1] Currently on leave with IBM, Thomas J. Watson Research Center, Yorktown Heights, NY 10598.

For each multiple access system (reservation and direct transmission) and for each feedback channel (binary and ternary), we introduce procedures that yield optimal *nested* multiple-access algorithms that are based on group testing techniques. The computational complexity and the memory needed to determine the optimal nested algorithms are exponential in the number of users, thus they are not practical for large or even moderate number of users.

Therefore, for large number of users, we introduce optimal *ordered* algorithms. Ordered algorithms are nested algorithms with an additional restriction. The users are ordered in some fixed order, say $u_1, u_2, \ldots, u_N$, that does not change during the cycle, and a user u_i cannot be classified (in a reservation system) or cannot transmit successfully (in a direct transmission system), before users $u_1, u_2, \ldots, u_{i-1}$. The computational complexity and the memory size needed to determine the optimal ordered algorithms are polynomial in the number of users.

We study the effect of various orderings on the performance of the algorithms and one of our interesting conclusions is that the performance of the various algorithms are not very sensitive to the different orderings as long as one employs "reasonable" orderings. Generally, in a reservation system one should order the users so that $p_1 \leq p_2 \leq \ldots \leq p_N$, while in a direct transmission system the order should be $p_1 \geq p_2 \geq \ldots \geq p_N$. Surprisingly, we found out that the performance of optimal ordered algorithms with the above orders do not fall very much behind the performance of optimal nested algorithms (at least for those cases that we could determine the optimal nested algorithms).

Finally, we also introduce several sub-optimal ordered algorithms whose determination is even simpler, and compare their performance with the performance of the optimal ordered algorithms.

References

[1] M. Sobel and P. A. Groll, "Group Testing to Eliminate Efficiently all Defectives in a Binomial Sample," Bell System Technical Journal, VOL. 38, pp. 1178-1252, Sept. 1959

[2] T. Berger, N. Mehravari, D. Towsly and J. K. Wolf, "Random Multiple- Access Communication and Group Testing," IEEE Trans. on Communications, COM-32, pp 769-778, July 1984.

[3] J. K. Wolf, "Born Again Group Testing: Multiaccess Communications," IEEE Trans. on Information Theory, Vol. IT-3125, pp. 185-191, March 1979.

SPLITTING-TYPE ALGORITHMS IN A MULTI-STATION PACKET-RADIO NETWORK

AYAL BAR-DAVID and MOSHE SIDI[1]

Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa 32000, Israel

SUMMARY

A new model for Packet Radio Networks (PRN) called the multi-station model, has been suggested by Cidon and Sidi in a recent paper [1]. A multi-station network consists of many nodes and several, say K, stations. All nodes share a slotted-time, collision type channel for data transmission. Each node has a predetermined station as the destination for its data, and follows feedback messages from that station, on a collision free channel (a dedicated feedback channel for each station). Due to the broadcast nature of the channel, a node may be heard by more than one station. Let $B(i)$ and $H(i)$ $1 \leq i \leq K$ be the set of nodes that transmit to station S_i, and the set of nodes that may be heard by S_i, respectively. Obviously, $B(i) \cap B(j) = \phi \;\; \forall\, i \neq j$, and $B(i) \subseteq H(i)$.

In [1], the region of feasible arrival rates has been determined, under the assumption that the nodes are following an ALOHA-type multi-access algorithm. The goal of this paper is to study the performance of a multi-station network whose nodes employ a splitting type multi-access algorithm [2].

Assume that the nodes of $B(i)$ follow the collision resolution algorithm (CRA) suggested by Capetanakis [3], by tracing the feedback messages of station S_i. Further, assume that the first time transmission rule is according to the epoch mechanism, described in [2]. Transmission of data packets by nodes in $H(i) - B(i)$ interferes with the CRA performed by the nodes of $B(i)$. Such interference may cause S_i to receive the following: (a) A collision instead of a successful transmission (if exactly one node of $B(i)$ and at least one node in $H(i) - B(i)$ transmit simultaneously);. denote this event by A_i. (b) A collision instead of an idle slot (if none of the nodes of $B(i)$, and at least two nodes of $H(i) - B(i)$ transmit);

---

[1] Currently on leave with IBM, Thomas J. Watson Research Center, Yorktown Heights, NY 10598.

denote this event by D_i. (c) A successful transmission instead of an idle slot (if none of the nodes of $B(i)$ and exactly one node of $H(i) - B(i)$ transmit). This event is not harmful since the receiving station may identify the successfully transmitting node as an outsider.

The behavior of a multi-station network employing a CRA may, in principle, be described as a Markov chain. The interaction among the resolution processes of nodes in $B(i)$ and nodes in $H(i) - B(i)$ leads, however, to a very complicated Markov chain, hence precise analysis of the network's performance is probably intractable. Rather, an approximation technique is presented in this paper for the analysis of the performance of such a network. The technique is based on interpreting the interference on station S_i by nodes of $H(i) - B(i)$ as channel errors.

Following is a short description of the approximation:

Let ε_i be the probability of event A_i, and δ_i the probability of event D_i. Our approximation is based on the assumption that the events A_i and D_i are independent of each other, and independent from slot to slot. With this assumption, one may derive equations for ε_j and δ_j in terms of ε_i and δ_i ($\forall\ i,\ \ i \neq j$), for all j, by following the evolution of the CRA. The set of $2K$ nonlinear equations in ε_i, δ_i $(1 \leq i \leq K)$ is solved numerically. Once the ε's and δ's are obtained, any performance measure such as the average delay, or the maximal throughput may be computed.

The results obtained using the approximation presented above, were found to be in good agreement with results obtained via simulations. For some extreme situations, such as no interference at all $(H(i) \cap H(j) = \phi \ \ \forall\ i \neq j)$, or complete interference $(H(i) = H(j) \ \ \forall\ i,j)$, one can derive exact values for the network performance. For these situations, our approximation yields results that are also in good agreement with these values.

References

[1] I. Cidon and M. Sidi, "A Single-Hop Multi-Station Packet-Radio Network", INFOCOM '84, pp. 336-343, San-Francisco, April 1984.

[2] R.G. Gallager, "A Perspective on Multiaccess Channels", IEEE Trans. on Information Theory, Vol. IT-31, No. 2, March 1985.

[3] J.I. Capetanakis, "Tree Algorithms for Packet Broadcast Channels", IEEE Trans. on Information Theory, Vol. IT-25, No. 5, Sept. 1979.

APPLICATION OF INTERVAL ANALYSIS TECHNIQUES TO THE PARAMETER TOLERANCE PROBLEM FOR LINEAR DYNAMICAL SYSTEMS

A. N. MICHEL*
Department of Electrical and
Computer Engineering
University of Notre Dame
Notre Dame, IN 46556
U.S.A.

E. P. OPPENHEIMER
Johns Hopkins University
Applied Physics Lab
Laurel, MD 20707
U.S.A.

ABSTRACT

In the present paper we explore the applicability of interval analysis techniques to issues of computing machine intervals which bound the set of all solutions associated with initial-value problems of linear systems of autonomous first-order ordinary differential equations that linearly depend on a parameter in an interval. The motivation for studying this problem includes many interesting applications. We cite here as examples worst case studies (in the time domain) of electric circuits (where components such as resistors, capacitors, etc. have tolerances associated with them), of optimal control problems, and the like.

In the final version of this paper we first address some fundamental properties of continuous and rational interval functions. We then use these results in studying the interval exponential matrix function. We finally demonstrate the applicability of our results in the analysis of several specific examples.

INTRODUCTION

In many practical problems one is interested in worst case studies (in the time domain) of dynamical systems endowed with parameters which are only known within given tolerances of specified nominal values. Specific examples which come immediately to mind are electric circuits (where components such as resistors, capacitors, inductors, etc., have tolerances associated with them), optimal control systems, and the like. In the complete version of the present paper we address problems of this type, using interval analysis methods [1], [2]. In our approach we avoid abstraction whenever possible. Indeed, the general mathematical tools employed are rather modest: they include some algebra (see, e.g., [3], [4]), some basic results from metric spaces (see, e.g., [5, Chapter 5], [6]) and some basic background in ordinary differential equations (see, e.g., [7]).

II. SUMMARY OF THE RESULTS

To be a little more specific than above, in the present paper we consider the applicability of interval analysis techniques to issues of com-

---

*The work of A. N. Michel was supported in part by the National Science Foundation under Grant ECS 84-19918.

puting machine intervals which bound the set of all solutions associated with initial-value problems of linear systems of autonomous first-order ordinary differential equations that linearly depend on a parameter in an interval. Towards this goal, we construct a metric space of continuous interval functions of an interval variable which is complete and includes the C[a,b] real function space and the "united extension" of its member functions f defined by $\bar{f}[a,b]) \overset{\Delta}{=} \bigcup_{x \varepsilon [a,b]} f([x,x])$. (The symbol [a,b] denotes the interval determined by $a \leq x \leq b$ and [a,a] denotes the degenerate interval determined by $a \leq x \leq a$.) We show that the rational interval functions belong to this space and exhibit the inclusion property of $f([a,b]) \supset \bar{f}([a,b])$ and that the sequence of partial sums obtained from the infinite series representation of the interval exponential function is found to be a Cauchy sequence, converging to a member function with this property. We devise a technique to compute an approximation $\tilde{g}$ of the interval exponential function $g([a,b]) \overset{\Delta}{=} \exp([a,b])$ which, for $\varepsilon > 0$, provides the error inclusive property $[1-\varepsilon, 1+\varepsilon] .g([a,b]) \supset \tilde{g}([a,b]) \supset g([a,b]) \supset \bar{g}([a,b])$. Defining a partitioning of the interval I = [a,b] by $I_i^n \overset{\Delta}{=} [a(n-i+1) + b(i-1), (n-i)a+ib]/n$, $i = 1, \ldots, n$, we establish the convergence result $g(I) \supset \lim_{n\to\infty} \bigcup_{i=1}^{n} g(I_i^n) \overset{\Delta}{=} \bar{g}(I)$. We then extend the above results to continuous interval matrix functions of an interval variable. This in turn enables us to obtain an approximation for the interval fundamental matrix and establish a convergence philosophy (employing the multiplicative property of the fundamental matrix) for the computation of interval bounds to the set of solutions for the parameterized initial-value problems.

We devise fast single-precision bounding interval arithmetic functions and we employ these in the development of a linear interval integration algorithm. We obtain algorithmic results (with reasonable computing efficiency) for parameterized examples of an RLC circuit, a fourth-order servomechanism, and a fifth-order minimum sensitivity optimal regulator design.

In [1], [2], Moore applies interval techniques to initial-value problems of nonlinear ordinary differential equations to determine numerical error inclusive bounds for solution trajectories; he does not consider trajectory bounds which also include the effects of a perturbation parameter in the differential equations. We emphasize that whereas our results are certainly related to existing works on interval analysis (see [1], [2] as well as the extensive bibliographies given in these references), to the best of our knowledge, the results of the present paper have not been reported elsewhere in form, scope or generality.

III. OUTLINE OF THE PAPER

Specific details concerning the structure of the final version of the present paper are as follows.

In the first section we will give an overview along the lines of the material given above.

In the second section we establish some basic notation and we demonstrate that interval arithmetic is not endowed with a rich algebraic structure.

In the third section we provide additional essential notation and we introduce a metric space as our appropriate mathematical setting.

In section four, we explore some pertinent properties of continuous interval functions.

In the fifth section we examine properties of rational interval functions.

In section six, we extend the results of sections 3, 4 and 5 to matrix interval functions.

In section seven, we use the results of sections 3, 4 and 5 to study the interval exponential function.

In section eight, we use the results of section 6 to study the properties of the interval matrix exponential function (resp., the interval fundamental matrix).

In section nine, we apply the results of the preceding sections in the study of initial-value problems.

We demonstrate the applicability of our results by considering in sections ten, eleven and twelve, three examples: a specific RLC circuit, a specific servomechanism and a specific optimal regulator design.

IV. SAMPLE EXAMPLES

In this section we summarize two sample examples to demonstrate the applicability of our results. The first of these is a specific example of a simple RLC circuit where R is assumed to have a very large tolerance associated with it while the second example is a practical instrument servomechanism with a load inertia parameter associated with it. In the final version of the paper we will also include an example which addresses a minimum plant sensitivity optimal linear regulator design.

Example 1

We consider an RLC circuit described by

$$L\ddot{q} + R\dot{q} + \frac{1}{C} q = Eu(t) , \qquad q(0) = 0, \ \dot{q}(0) = 0$$

where u(t) denotes the unit step. We assume that

$$L = 1.0, \ C = 1.0, \ E = 1.0 \text{ and}$$

$$R = 0.4 \pm 0.2 .$$

Then the normalized RLC circuit equation assumes the form

$$\ddot{x} + 2\delta \ \omega_n \dot{x} + \omega_n{}^2 x = u(t)$$

where $\omega_n = 1.0$ and

$$\delta = 0.2 \pm 0.1 .$$

Letting $x_1 = q$, $\dot{x}_1 = x_2$, $\dot{x}_3 = 0$, $x_3 = Eu(t)$, and letting θ denote the value of R in the "tolerance" interval [0.2, 0.6], the RLC circuit is described by the system of equations

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & -\theta & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} , \qquad \begin{bmatrix} x_1(0) \\ x_2(0) \\ x_2(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\theta \in [0.2, 0.6] .$$

Figure 1 depicts the interval bound solutions obtained by our algorithm. Specifically, $x_1(t)$ and $x_2(t)$ will be within the indicated envelopes for any θ in the interval [0.2, 0.6], resp., for any R in the interval [0.2, 0.6].

Example 2.

This example is encountered in a practical design of an instrument servomechanism with load inertia parameter [8]. The nonlinear model of this system is depicted in Fig. 2. A linearized version of this system is obtained by (i) replacing the coulomb friction by viscons damping f_L, (ii) setting the motor starting voltage equal to zero, (iii) assuming that the amplifier saturation limits are sufficiently large so that they may be neglected, and (iv) removing the backlash deadzone.

Let $x_1 = \dot{\theta}_m$, $x_2 = \theta_m$, $x_3 = \dot{\theta}_L$, $x_4 = \theta_L$, and $x_5 = \theta_i$ and assume that the follow-up, generator and amplifier carrier phase shifts are each zero. Also, assume a unit step positional displacement input. Let $x^T = (x_1, x_2, x_3, x_4, x_5)$. Then the instrument servo may be described by the system of equations

$$\underline{\dot{x}} = \begin{bmatrix} -\dfrac{B_m + K_q K_a K_m}{J_m} & -\dfrac{K_s}{N^2 J_m} & 0 & \dfrac{K_s - K_f K_a N K_m}{N^2 J_m} & \dfrac{K_f K_a K_m}{N J_m} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & \dfrac{K_s}{J_L} & -\dfrac{f_L}{J_L} & -\dfrac{K_s}{J_L} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \underline{x}$$

and

$$\underline{x}(0) = (0\ 0\ 0\ 0\ 1)^T .$$

The following coefficient values are given in [8, p. 103], with the exception of the tolerance on the load inertia:

$$K_f = 18.0,\ K_m = 0.0026,\ K_a = 240,\ K_g = 0.14,$$

$$K_s = 1500,\ B_m = 0.0002,\ N = 100,\ J_m = 1.0,\ f_L = 0.0$$

$$J_L = 3000.0 \pm 300 .$$

The interval algorithm results for this example are shown in Figures 3 and 4 where the discrete time interval solution bounds obtained from the algorithm are plotted. Superimposed on each plot are continuous solution curves for the nominal and endpoint values of the load inertia parameters.

REFERENCES

[1] R.E. Moore, Interval Analysis, Prentice-Hall, Englewood Cliffs, 1966.

[2] R.E. Moore, Methods and Applications of Interval Analysis, SIAM Studies in Applied Mathematics, Philadelphia, 1979.

[3] N. Jacobsen, Lectures in Abstract Algebra, Van Nostrand, New York, 1951.

[4] C.B. Hanneken, Introduction to Abstract Algebra, Dickenson Publishing Co., Belmont, CA, 1968.

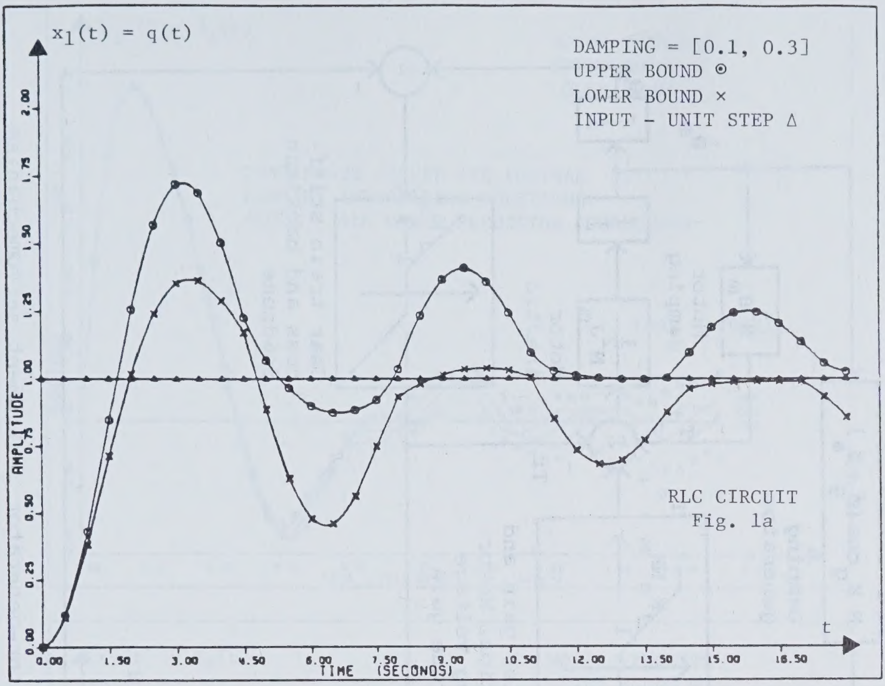
[5] A.N. Michel and C.J. Herget, Mathematical Foundations in Engineering and Science: Algebra and Analysis, Prentice-Hall, Englewood Cliffs, 1981.

[6] A. W. Naylor and G. R. Sell, Linear Operator Theory in Engineering and Science, Holt, Rinehart and Winston, New York, 1971.

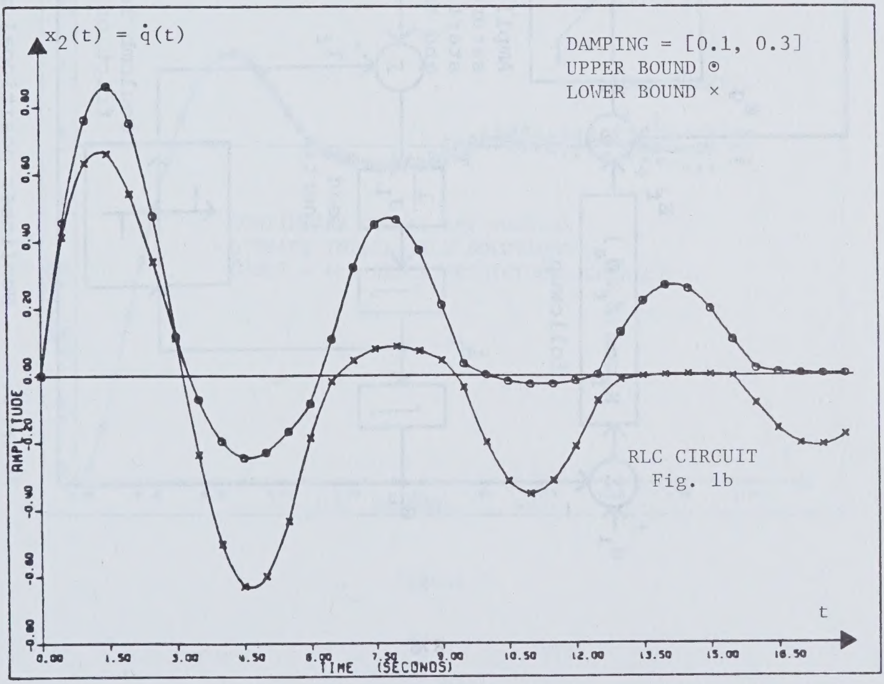
[7] R.K. Miller and A.N. Michel, Ordinary Differential Equations, Academic Press, New York, 1982.

[8] B.A. Chubb, Modern Analytical Design of Instrument Servomechanisms, Addison Wesley Publishing Co., Reading, MA, 1967.

[9] Proceedings of an International Symposium on Interval Mathematics, Institut für angewandte Mathematik, Universität Freiburg, West Germany, May 27-31, 1980, Academic Press, NY, 1980.



RLC CIRCUIT
Fig. 1a



RLC CIRCUIT
Fig. 1b

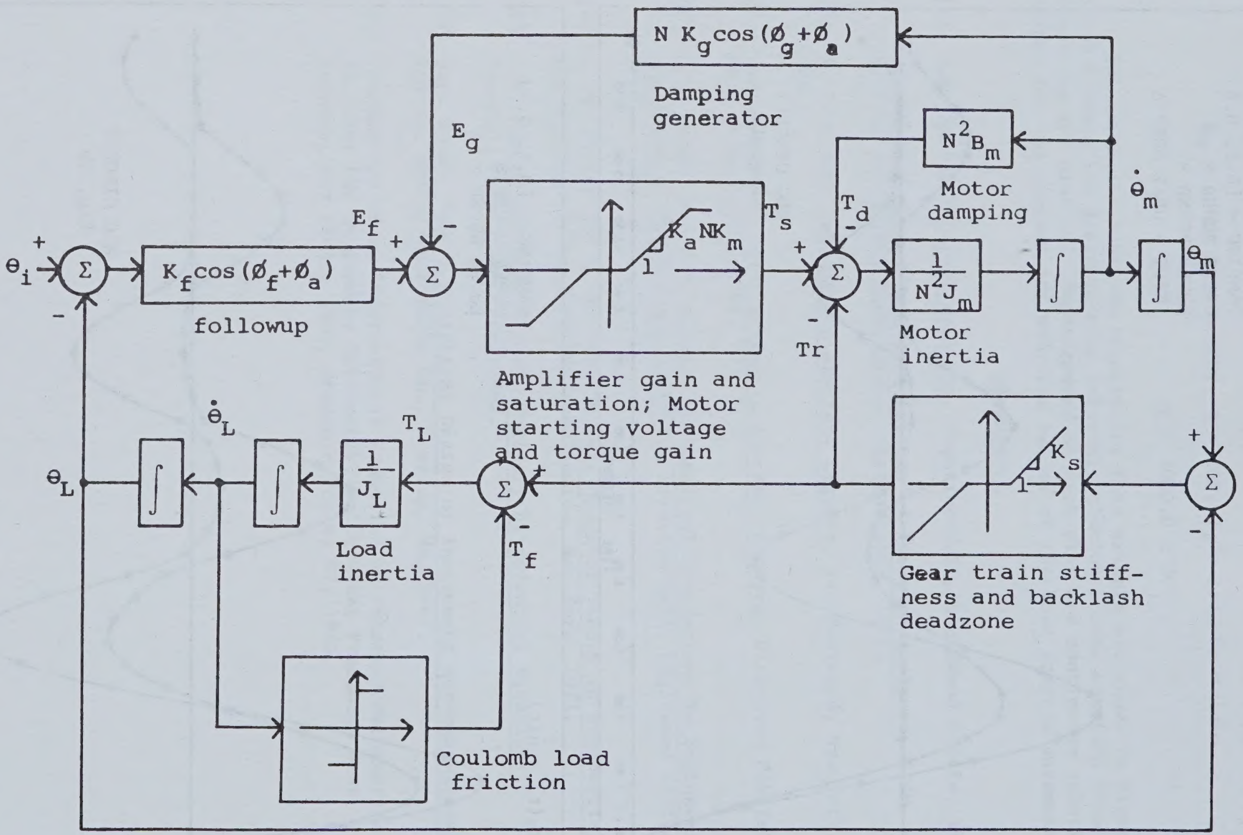

θ_i
+
-
Σ
$K_f \cos(\emptyset_f+\emptyset_a)$
followup
E_f
+
-
E_g
$N\ K_g \cos(\emptyset_g+\emptyset_a)$
Damping generator
$K_a NK_m$
1
T_s
+
T_d
-
Tr
N^2B_m
Motor damping
$\frac{1}{N^2J_m}$
Motor inertia
$\dot{\theta}_m$
θ_m
Amplifier gain and saturation; Motor starting voltage and torque gain
K_s
Gear train stiffness and backlash deadzone
T_L
$\frac{1}{J_L}$
Load inertia
$\dot{\theta}_L$
θ_L
T_f
Coulomb load friction

Figure 2. Nonlinear positional motor-generator instrument servomechanism

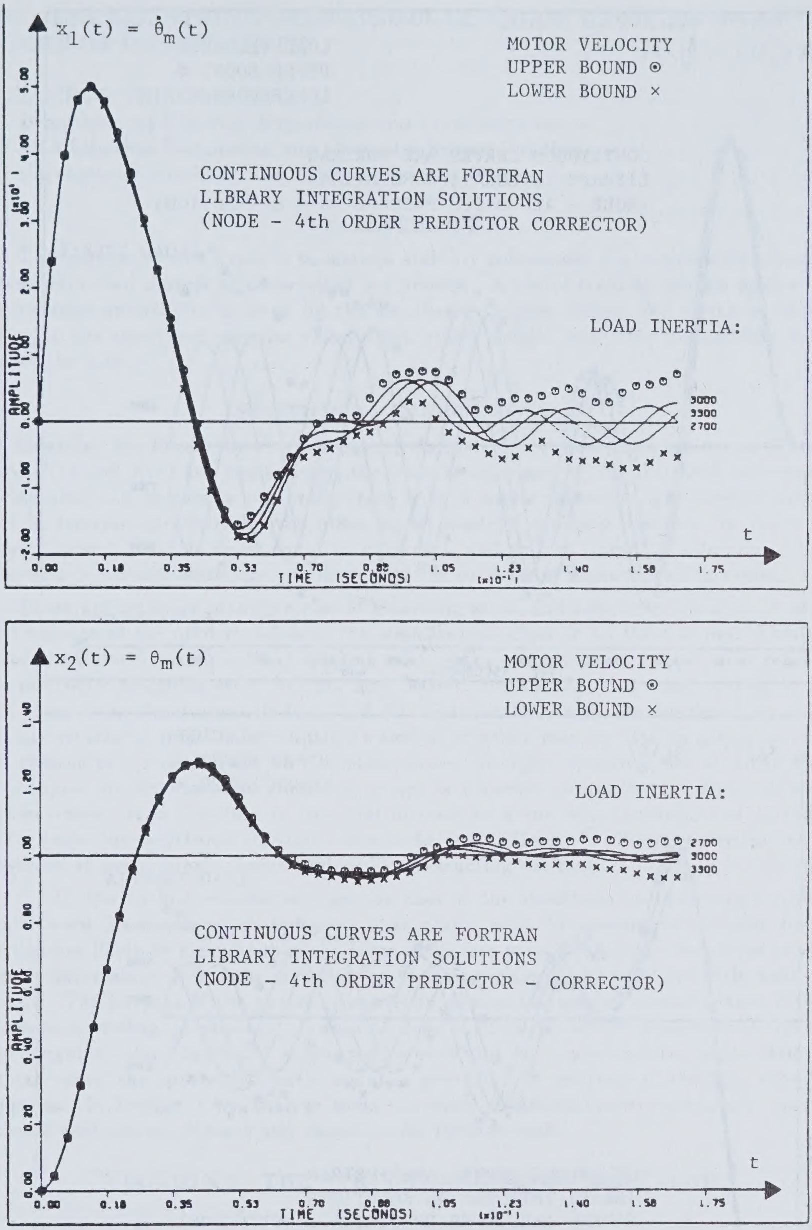
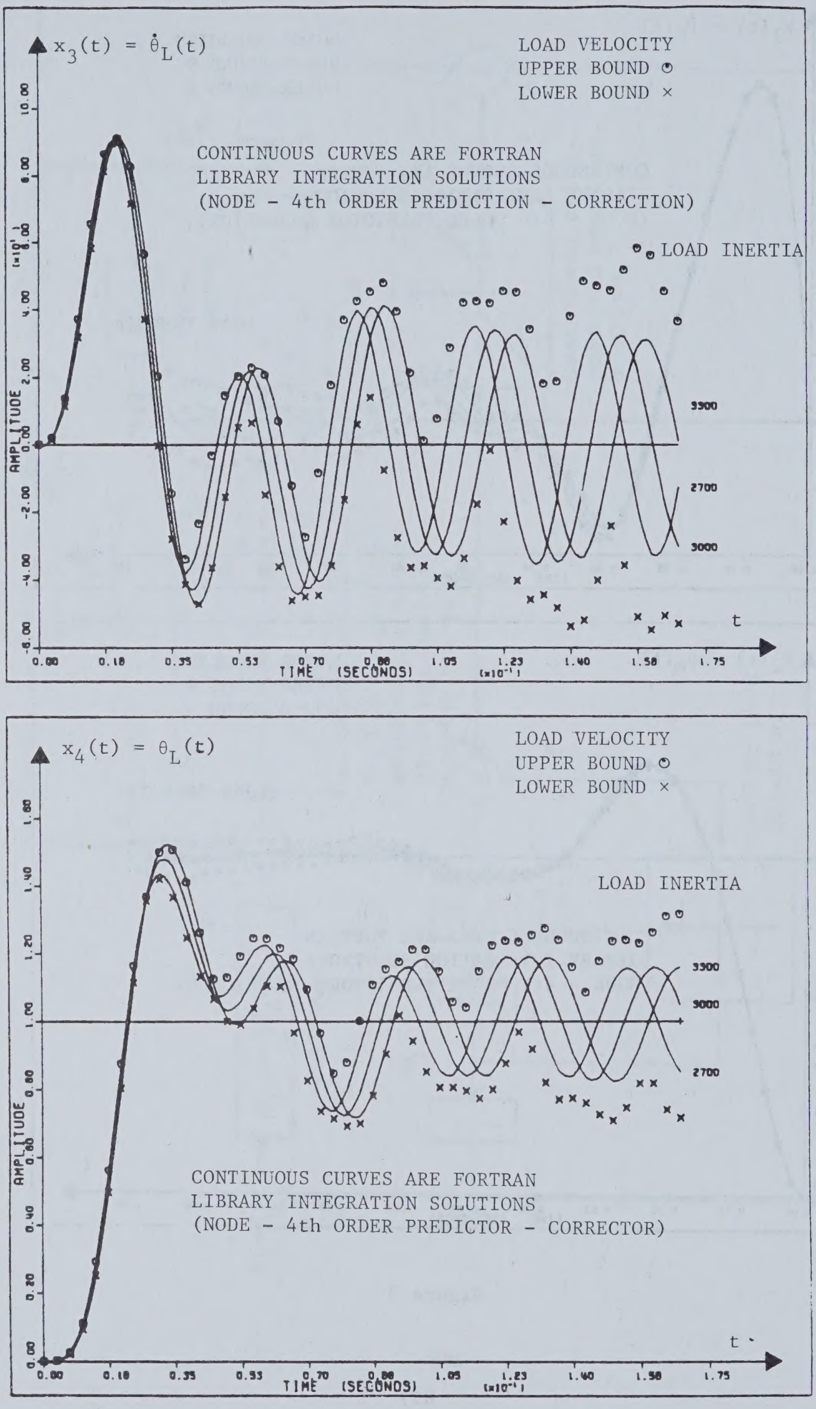

$x_1(t) = \dot{\theta}_m(t)$
MOTOR VELOCITY
UPPER BOUND ⊙
LOWER BOUND ×
CONTINUOUS CURVES ARE FORTRAN
LIBRARY INTEGRATION SOLUTIONS
(NODE - 4th ORDER PREDICTOR CORRECTOR)
LOAD INERTIA:
3000
3300
2700
AMPLITUDE
(x10^1)
5.00
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
0.00
0.18
0.35
0.53
0.70
0.88
1.05
1.23
1.40
1.58
1.75
TIME (SECONDS)
(x10^-1)
t
$x_2(t) = \theta_m(t)$
MOTOR VELOCITY
UPPER BOUND ⊙
LOWER BOUND ×
LOAD INERTIA:
2700
3000
3300
CONTINUOUS CURVES ARE FORTRAN
LIBRARY INTEGRATION SOLUTIONS
(NODE - 4th ORDER PREDICTOR - CORRECTOR)
AMPLITUDE
1.40
1.20
1.00
0.80
0.60
0.40
0.20
0.00
0.00
0.18
0.35
0.53
0.70
0.88
1.05
1.23
1.40
1.58
1.75
TIME (SECONDS)
(x10^-1)
t

Figure 3



$x_3(t) = \dot{\theta}_L(t)$
LOAD VELOCITY
UPPER BOUND ⊙
LOWER BOUND ×
CONTINUOUS CURVES ARE FORTRAN
LIBRARY INTEGRATION SOLUTIONS
(NODE - 4th ORDER PREDICTION - CORRECTION)
LOAD INERTIA
3300
2700
3000
t
AMPLITUDE
(×10¹)
-6.00
-4.00
-2.00
0.00
2.00
4.00
6.00
8.00
10.00
0.00
0.18
0.35
0.53
0.70
0.88
1.05
1.23
1.40
1.58
1.75
TIME (SECONDS)
(×10⁻¹)
$x_4(t) = \theta_L(t)$
LOAD VELOCITY
UPPER BOUND ⊙
LOWER BOUND ×
LOAD INERTIA
3300
3000
2700
CONTINUOUS CURVES ARE FORTRAN
LIBRARY INTEGRATION SOLUTIONS
(NODE - 4th ORDER PREDICTOR - CORRECTOR)
t
AMPLITUDE
0.00
0.20
0.40
0.60
0.80
1.00
1.20
1.40
1.60
0.00
0.18
0.35
0.53
0.70
0.88
1.05
1.23
1.40
1.58
1.75
TIME (SECONDS)
(×10⁻¹)

Figure 4

THE GENERAL STRUCTURED SINGULAR VALUE PROBLEM WITH TWO BLOCKS OF UNCERTAINTY

JAMES S. FREUDENBERG
Department of Electrical Engineering and Computer Science
4213 Electrical Engineering and Computer Science Building
University of Michigan, Ann Arbor, Michigan 48109

ABSTRACT

The purpose of this paper is to analyze stability robustness of a multivariable feedback system when two sources of uncertainty are present. A useful stability margin against such simultaneous uncertainty is given by the structured singular value. We derive an approximation to the structured singular value which yields insight into when the stability robustness will be poor.

SECTION 1: INTRODUCTION

Consider the linear time-invariant multivariable feedback system shown in Figure 1, where $P(s)$ and $F(s)$ are, respectively, the transfer functions of the plant and compensator. We say that this system is *nominally stable* if it is stable whenever the true-but-unknown plant is described precisely by the plant model used for stability analysis. In practice, of course, the true plant is never equal to its model, and we are forced to determine whether the system is *robustly stable;* i.e., stable despite the presence of plant modelling errors.

There are typically many sources of modelling error, and robustness analysis is complicated because of the need to consider the *simultaneous* effect of all these errors. This problem is particularly vexing in that systems exist which are extremely robust with respect to each *individual* modelling error, but yet may become unstable for very small modelling errors when these occur simultaneously (e.g.,[1, 2, 3]). Fortunately, Doyle has developed an analysis tool, the *structured singular value* ([2, 3]) which in principle may be used to assess robustness with respect to a wide variety of modelling errors. Roughly speaking, the stability margin with respect to simultaneous modelling errors is inversely proportional to the structured singular value. It is therefore of practical interest to know what properties of a feedback system cause the structured singular value to be large. Hopefully, these properties can then be avoided in performing a design and/or in constructing the plant which is to be controlled.

In [4], the author considered a special case of the simultaneous uncertainty problem. Results were presented which indicated that plants with ill-conditioned transfer function matrices are likely to yield feedback systems which are very sensitive to the effects of simultaneous uncertainty unless the structure of the ill-conditioned plant is explicitly taken into account. The purpose of the present paper is to discuss the general case of plants with two sources of modelling uncertainty. In Section 2 we shall briefly review properties of the structured singular value. Section 3 is devoted to obtaining an approximation to the structured singular value; the approximation is useful in providing an estimate of stability robustness properties. In Section 4 we analyze some examples of simultaneous uncertainty problems. Section 5 contains conclusions and directions for further work.

SECTION 2: THE STRUCTURED SINGULAR VALUE

---

*This research was supported by NSF Grant ECS 8504558 and University of Michigan Rackham Faculty Grant No. 387994.

The purpose of this section is to briefly review those properties of the structured singular value needed in this paper. Space limitations dictate that we omit many useful properties of this function. For a complete discussion, see ([2, 3]).

In order to perform structured singular value analysis, it is first necessary to rearrange the block diagram (Figure 1) into an equivalent form (Figure 2). This configuration has the property that the uncertain elements, or *perturbations,* appear in a block diagonal matrix $\Delta(s) \in \mathbf{D}(s)$, where $\mathbf{D}(s) \overset{\Delta}{=} \{diag\{\Delta_i(s)\}\colon \Delta(s) \in \mathcal{C}^{k_i \times k_i}\}$. The matrix $M(s)$, which represents the interactions of the feedback system with the plant perturbations, is stable if the system is nominally stable. (Examples of various $M(s)$ matrices are found in Section 4.)

It is often assumed that each of the perturbations $\Delta_i(s)$ is an arbitrary member of a set of stable matrix transfer functions satisfying a frequency dependent norm bound. That is, assume that $\Delta_i(s) \in \mathbf{D}_i(s)$, where

$$\mathbf{D}_i(s) \overset{\Delta}{=} \{\ \Delta_i(s) : \ \Delta_i(s) \in \mathcal{C}^{k_i \times k_i} \text{ is stable and } \bar{\sigma}[\Delta_i(j\omega)] \leq M_i(\omega)\ , \quad \forall\ \omega\ \} \quad ,(2.1)$$

and $\bar{\sigma}[\cdot]$ is the *largest singular value*[5]. An uncertainty satisfying (2.1) will be termed an *unstructured perturbation.* In general, additional information about $\Delta_i(s)$ will be available. In some cases it is possible to incorporate this information into the problem statement using weighting functions[3]. For notational simplicity, we omit discussion of these, although it is straightforward to incorporate weighting functions into the results of this paper.

Suppose that the feedback system in Figure 1 is nominally stable, and let the system be rearranged to isolate the uncertainty as in Figure 2. The original system is stable for all $\Delta(s)$ allowed by (2.1) if and only if the rearranged system has this property. Since nominal stability is assumed, it follows ([2, 3]) that stability robustness will be present if and only if the transfer function $[I + M(s)\,\Delta(s)]^{-1}$ has no unstable poles for all $\Delta(s)$ allowed by (2.1). Hence stability robustness can be measured by determining, at each frequency, the smallest perturbation[1] for which $\det[I + M\Delta] = 0$.

Stability robustness may be measured using the structured singular value, which is defined[2] as follows:

$$\mu[M] = \begin{cases} 0\ , & \text{if no } \Delta \in \mathbf{D} \text{ exists with } \det[I+M\Delta]=0 \\ \left[\min\limits_{\Delta \in \mathbf{D}}\{\bar{\sigma}[\Delta]:\det(I+M\Delta)=0\}\right]^{-1}\ , & \text{otherwise}\ . \end{cases} \quad (2.2)$$

Existence of a small matrix $\Delta \in \mathbf{D}$ for which $\det[I+M\Delta] = 0$ therefore corresponds to a large value of $\mu[M]$, and vice versa. The structured singular value is thus an indication of the system's stability margin. Indeed, if it happens that

$$\mu[M(j\omega)] < \alpha(\omega)\ , \quad \forall\ \omega\ , \quad (2.3)$$

then the system will be robustly stable for all uncertainty of the form (2.1) with

$$M_i(\omega) = 1/\,\alpha(\omega) \quad \forall\ i \text{ and } \forall\ \omega\ . \quad (2.4)$$

When more than one perturbation is present, the matrix Δ contains more than one diagonal block. Suppose that we analyze the uncertainties one at a time; i.e., assume that $\Delta_j(s) = 0$, $j \neq i$. Then the stability margin with respect to the individual uncertainty

---

[1] When we deal with matrices evaluated at a fixed frequency, we shall suppress dependence on the frequency variable.

[2] Note that the structured singular value depends upon M as well as upon the structure of the uncertainty matrix Δ, by which we mean the number of blocks in Δ together with the size of each block. However, it is standard to only indicate the dependence upon M in the notation.

$\Delta_i(s)$ is given by the well-known singular value robustness test[5]: the system is robustly stable for all Δ_i of the form (2.1) with

$$M_i(\omega) < 1/\bar{\sigma}[M_{ii}(j\omega)]\ , \quad \forall\ \omega\ . \tag{2.5}$$

It follows that the size of each diagonal block of M is a lower bound on $\mu[M]$. As we pointed out in the Introduction, systems exist which are robust with respect to large perturbations, when taken one at a time, but are also extremely sensitive to small perturbations occurring simultaneously (e.g., see[1, 2, 3]). The importance of the structured singular value is that it allows one to test for the existence of this situation.

SECTION 3: A SPECIAL CASE: TWO BLOCKS OF UNCERTAINTY

We saw in the preceding section that the structured singular value is, by definition, a useful tool with which to measure the stability margin with respect to simultaneous uncertainty. We also remarked that the stability margins with respect to perturbations acting one at a time (i.e., the $\bar{\sigma}[M_{ii}(j\omega)]$) may not be an accurate indication of the stability margin with respect to simultaneous perturbations (i.e., $\mu[M(j\omega)]$). It should be clear, however, that it is the *off-diagonal* blocks of M which are responsible for the robustness difficulties caused by simultaneous perturbations. Indeed, if the off-diagonal blocks are zero, then $\mu[M] = \max_i \bar{\sigma}\,[M_{ii}]$. Our purpose in this section is to analyze the effect of these off-diagonal blocks in the special case that only two perturbations are present.

Denote the uncertainty matrix by $\Delta = diag\,[\Delta_1,\Delta_2]$ and partition the matrix M as

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}\ , \tag{3.1}$$

where $M_{11} \in \mathcal{C}^{\,k_1 \times k_1}$ and $M_{22} \in \mathcal{C}^{\,k_2 \times k_2}$, where κ_1 and κ_2 are the dimensions of the blocks Δ_1 and Δ_2, respectively. Our analysis will be based upon the characterization (2.2) of the structured singular value. Hence we shall attempt to estimate the smallest simultaneous perturbation for which the determinant

$$\begin{aligned} \det[I + M\Delta] &\equiv \det \begin{bmatrix} I + M_{11}\Delta_1 & M_{12}\Delta_2 \\ M_{21}\Delta_1 & I + M_{22}\Delta_2 \end{bmatrix} \\ &= 0\ . \end{aligned} \tag{3.2}$$

If the uncertainty $\Delta_1 = 0$, then the stability margin with respect to the uncertainty Δ_2 is given by $\bar{\sigma}[M_{22}]$. Hence, with $\Delta_1(s) = 0$, the system is robustly stable for all Δ_2 of the form (2.1) with

$$M_2(\omega) < 1/\bar{\sigma}\,[M_{22}(j\omega)]\ , \quad \forall\ \omega\ . \tag{3.3}$$

This observation suggests that the simultaneous uncertainty problem be viewed as one of determining the effect that the perturbation Δ_1 has upon the stability margin with respect to Δ_2. (Obviously, we could ask the same questions with the roles of Δ_1 and Δ_2 reversed.) The following lemma yields a factorization of (3.2) which will prove to be useful in these tasks. The proof is a straightforward application of a well-known formula for the determinant of a partitioned matrix.

Lemma 3.1: Consider the determinant (3.2), with matrix M defined by (3.1). If $\det[I + M_{11}\Delta_1] \neq 0$, then

$$\det[I + M\Delta] = \det[I + M_{11}\Delta_1] \cdot \det\{\, I + [M_{22} - M_{21}\Delta_1(I + M_{11}\Delta_1)^{-1}M_{12}]\Delta_2\} \tag{3.4}$$

■

An analogous factorization may be obtained if $\det[I + M_{22}\Delta_2] \neq 0$. For purposes of brevity, we shall omit discussion of this factorization.

The factorization (3.4) shows that when the uncertainty $\Delta_1 \neq 0$, the stability margin at each frequency with respect to the perturbation Δ_2 becomes $\overline{\sigma}[\widetilde{M}_{22}(\Delta_1)]$, where

$$\widetilde{M}_{22}(\Delta_1) \triangleq M_{22} - M_{21}\Delta_1(I + M_{11}\Delta_1)^{-1}M_{12} \quad . \tag{3.5}$$

Suppose that a *small* perturbation Δ_1 exists for which $\overline{\sigma}[\widetilde{M}_{22}(\Delta_1)]$ is *large*. Then it follows from well-known properties of singular values that a *small* perturbation Δ_2 exists such that

$$\overline{\sigma}[\Delta_2] = 1/\,\overline{\sigma}[\widetilde{M}_{22}(\Delta_1)] \tag{3.6}$$

and $\det[I + M\Delta] = 0$.

We are especially interested in the case that small *simultaneous* perturbations can cause instability in a system which is robust with respect to large *individual* perturbations. Hence it makes sense to consider only the *first-order* effects of Δ_1 upon $\widetilde{M}_{22}(\Delta_1)$. If the first order approximation to $\widetilde{M}_{22}(\Delta_1)$ reveals that this matrix is very sensitive to small deviations in Δ_1 from its nominal value of zero, then this is an indication that stability robustness problems may exist.

Lemma 3.2: Consider the matrix $\widetilde{M}_{22}(\Delta_1)$ defined by (3.5). This matrix may be approximated, to first order in Δ_1, by the matrix

$$\widetilde{M}_{22}^{app}(\Delta_1) = M_{22} - M_{21}\Delta_1 M_{12} \quad . \tag{3.7}$$

■

Proof: Using the identity

$$(I + M_{11}\Delta_1)^{-1} = I - M_{11}\Delta_1(I + M_{11}\Delta_1)^{-1} \tag{3.8}$$

in (3.5) yields

$$\widetilde{M}_{22}(\Delta_1) = M_{22} - M_{21}\Delta_1 M_{12} + M_{21}\Delta_1 M_{11}\Delta_1(I + M_{11}\Delta_1)^{-1}M_{12} \quad , \tag{3.9}$$

from which (3.7) follows.

■

From (3.7) it follows that the stability margin with respect to the perturbation Δ_2 will be very sensitive with respect to the perturbation Δ_1 unless the product

$$\overline{\sigma}[M_{12}]\;\overline{\sigma}[M_{21}] \tag{3.10}$$

is sufficiently small. This observation suggests that design goals should include the requirement that (3.10) be as small as possible. Unfortunately, requiring (3.10) to be small may conflict with other design goals, such as the requirements that $\overline{\sigma}[M_{11}]$ and/or $\overline{\sigma}[M_{22}]$ be small. The latter requirements are necessary to achieve an adequate stability margin with respect to each perturbation when the other is zero.

The potential design tradeoff just described motivates us to seek a quantitative estimate of the effect that nonzero off-diagonal blocks of M have upon the stability margin with respect to simultaneous perturbations. Toward this end, recall the factorization (3.4), which showed that $\det[I + M\Delta]$ will become singular if

$$\det[I + \widetilde{M}_{22}(\Delta_1) \cdot \Delta_2] = 0 \tag{3.11}$$

It seems reasonable to estimate the smallest perturbation for which (3.11) holds by calculating the smallest perturbation for which the first order approximation $\widetilde{M}_{22}^{app}(\Delta_1)$ satisfies

$$\det[I + \widetilde{M}_{22}^{app}(\Delta_1) \cdot \Delta_2] = 0 \quad . \tag{3.12}$$

If this condition holds, then it is straightforward to verify that $I + \widetilde{M}_{22}(\Delta_1) \cdot \Delta_2$ is singular except for terms which are at least second order in Δ_1. Hence, if one can find *small* perturbations Δ_1 and Δ_2 for which (3.12) holds, then it is plausible to expect that the size of $\Delta = diag[\Delta_1, \Delta_2]$ would approximate the size of the minimum destabilizing perturbation in definition (2.2). By analogy with (2.2), we define

$$\mu_2^{app}[M] \triangleq \begin{cases} 0 \;, & \text{if no } \Delta \text{ exists such that (3.12) is satisfied} \\ \\ \left[\min\limits_{\Delta}\{\overline{\sigma}[\Delta]\text{: (3.12) holds}\}\right]^{-1} \;, & \text{otherwise} \quad . \end{cases} \tag{3.13}$$

The approximation (3.13) to the structured singular value can be calculated easily using a corollary to the following lemma. We shall need the following notation. Suppose that the matrix M_{12} has a left inverse. Then there exists a matrix M_{12}^L such that $M_{12}^L M_{12} = I$. Similarly, if M_{21} has a right inverse, there exists a matrix M_{21}^R such that $M_{21}M_{21}^R = I$.

Lemma 3.3: Assume that the matrix M_{12} has a left inverse and that M_{21} has a right inverse.[3] Then, for each $\alpha_1 \geq 0$, and

$$\alpha_2 \triangleq 1 \Big/ \Big(\overline{\sigma}[M_{22}] + \overline{\sigma}[M_{21}]\overline{\sigma}[M_{12}]\alpha_1\Big) \tag{3.14}$$

there exist matrices Δ_1 and Δ_2 such that $\overline{\sigma}[\Delta_1] = \alpha_1$, $\overline{\sigma}[\Delta_2] = \alpha_2$, and (3.12) holds.

■

Proof: Denote the left and right singular vectors corresponding to the largest singular value of M_{22} by $\overline{v}$ and $\overline{u}$, respectively. Define $\Delta_1 \triangleq c_1 M_{21}^R \overline{vu}^H M_{12}^L$, where $c_1 = -\alpha_1\overline{\sigma}[M_{21}]\overline{\sigma}[M_{12}]$. Then $M_{21}\Delta_1 M_{12} = -\alpha_1\overline{\sigma}[M_{21}]\overline{\sigma}[M_{12}]\overline{vu}^H$, and thus $\overline{\sigma}[M_{22} - M_{21}\Delta_1 M_{12}] = \overline{\sigma}[M_{22}] + \overline{\sigma}[M_{21}]\overline{\sigma}[M_{12}]\alpha_1 = 1/\alpha_2$. It follows from well-known properties of singular values that there exists a matrix Δ_2 with $\overline{\sigma}[\Delta_2] = 1/\alpha_2$ and $\det[I + (M_{22} - M_{21}\Delta_1 M_{12})\Delta_2] = 0$.

■

Given the size of the perturbation Δ_1, this lemma is useful in that it allows us to determine the size of the smallest perturbation Δ_2 such that (3.12) holds. Defining

$$\alpha_{\min} \triangleq \min_{\alpha_1} \max\{\alpha_1, \alpha_2\text{: (3.14) holds}\} \quad ,$$

it follows that the smallest Δ for which (3.12) holds has size $\overline{\sigma}[\Delta] = \alpha_{\min}$. To determine $\alpha_{\min}$, it is useful to sketch a locus of the points (α_1, α_2) satisfying (3.14). Inspection of such a curve reveals that $\alpha_{\min}$ is achieved at the point for which $\alpha_1 = \alpha_2$. These observations yield

---

[3] This assumption will usually be satisfied when $k_1 \geq k_2$. If $k_1 < k_2$, then one can obtain an analogous

Corollary 3.4: The approximation (3.13) to the structural singular value is equal to

$$\mu_2^{app}[M] = 1/\,\alpha_{\min} \quad , \tag{3.15}$$

where

$$\alpha_{\min} = \frac{-\,\bar\sigma[M_{22}] + \sqrt{\bar\sigma[M_{22}]^2 + 4\bar\sigma[M_{12}]\bar\sigma[M_{21}]}}{2\bar\sigma[M_{12}]\bar\sigma[M_{21}]} \quad . \tag{3.16}$$

■

Proof: Set $\alpha_1 = \alpha_2 = \alpha_{\min}$ in (3.14) and solve the resulting quadratic equation.

■

Recall our comment after Lemma 3.1 that a factorization analogous to (3.4) could be obtained with the roles of M_{22} and M_{11} reversed. Using this factorization, one can show that a result identical to (3.16) holds, with the exception that M_{22} is replaced by M_{11}.

Using (3.16) it is possible to obtain insight into the effect that the off-diagonal blocks M_{12} and M_{21} have upon stability robustness. Note in particular that, as the ratio $\bar\sigma[M_{22}]^2 \,/\, \bar\sigma[M_{12}] \cdot \bar\sigma[M_{21}] \to 0$,

$$\alpha_{\min} \to 1 \,/\, \sqrt{\bar\sigma[M_{12}]\bar\sigma[M_{21}]} \quad . \tag{3.17}$$

Hence (3.13), the approximation to the structured singular value, satisfies

$$\mu_2^{app}[M] \to \sqrt{\bar\sigma[M_{12}]\bar\sigma[M_{21}]} \quad . \tag{3.18}$$

Clearly, if the product of the off diagonal blocks is too large, the system will not be robust with respect to simultaneous uncertainties. This is consistent with our earlier observation that it is these terms which can couple the effects of the two perturbations to cause a lack of robustness with respect to both simultaneously, even though robustness is good with respect to each alone. It is also straightforward to show that, as the ratio $\bar\sigma[M_{12}]\bar\sigma[M_{21}] / \,\bar\sigma[M_{22}]^2 \to 0$,

$$\mu_2^{app}[M] \to \bar\sigma\,[M_{22}] \quad . \tag{3.19}$$

Again, this limiting behavior is expected since $\bar\sigma[M_{22}]$ is the stability margin with respect to the perturbation Δ_2 when $\Delta_1 = 0$.

To conclude this section, we note that if $M_{11} = M_{22} = 0$, then

$$\begin{aligned} \det[I + M\Delta] &= \det \begin{bmatrix} I & M_{12}\Delta_2 \\ M_{21}\Delta_1 & I \end{bmatrix} \\ &= \det \Big[I - M_{21}\Delta_1 M_{12}\Delta_2 \Big] \end{aligned} \tag{3.20}$$

It is easy to show that, in this case,

$$\mu[M] = \sqrt{\bar\sigma[M_{12}]\bar\sigma[M_{21}]} \quad . \tag{3.21}$$

To see this, let $M_{21}^{\#}$ and $M_{12}^{\#}$ denote the pseudo inverses of M_{21} and M_{12}. Let $\Delta_1 = c_1 M_{21}^{\#} M_{12}^{\#}$, where $c_1 = \alpha_1 \bar\sigma\,[M_{12}]\bar\sigma\,[M_{21}]$. Then $\bar\sigma\,[\Delta_1] = \alpha_1$ and $\bar\sigma\,[M_{21}\Delta_1 M_{12}] = \alpha_1 \,\bar\sigma\,[M_{12}]\bar\sigma\,[M_{21}]$. Standard results show that there exists Δ_2 with $\bar\sigma\,[\Delta_2] = 1/(\alpha_1 \bar\sigma\,[M_{12}]\,\bar\sigma\,[M_{21}])$ and for which (3.20) equals zero. The result (3.21) follows easily. Hence, in this case, the approximation (3.18) is *exact.*

---

result by reversing the roles of M_{11} and M_{22} in Lemmas 3.1-3.3.

SECTION 4: STANDARD UNCERTAINTY MODELS

There are several descriptions of plant modelling error, of the general type discussed in the present paper, which have appeared extensively in the literature. A list of these error description, along with the type of uncertainty characterized by each, is found in [3]. When any of these errors appears by itself, a standard singular value test suffices to check for stability robustness. When two or more of these errors appear simultaneously, the structured singular value must be used to determine robustness properties. Our purpose in this section is to analyze the simultaneous uncertainty problem when *two* of these uncertainty sources are present.

Define the *open loop* transfer functions at the plant input and output by $L_I(s) \triangleq F(s)P(s)$ and $L_O(s) \triangleq P(s)F(s)$, respectively. Let the *sensitivity* and *complementary sensitivity* functions at the plant input be defined by $S_I(s) \triangleq [I + L_I(s)]^{-1}$ and $T_I(s) = L_I(s)[I + L_I(s)]^{-1}$, respectively, and let the same functions defined at the plant output be denoted $S_O(s) \triangleq [I + L_O(s)]^{-1}$ and $T_O(s) \triangleq L_O(s)[I + L_O(s)]^{-1}$.

Suppose that uncertainty is present simultaneously at both the plant input and output, possibly due to errors in both the sensor and actuator models. To illustrate, consider the following uncertainty models with associated matrices M and Δ:

P'	Δ	M	
$(I + \Delta_2)^{-1}P(I + \Delta_1)^{-1}$	$diag\,[\Delta_1,\Delta_2]$	$\begin{bmatrix} S_I & -P^{-1}T_O \\ S_OP & S_O \end{bmatrix}$	(4.1)
$(I + \Delta_2)^{-1}P(I + \Delta_3)$	$diag\,[\Delta_3,\Delta_2]$	$\begin{bmatrix} T_I & P^{-1}T_O \\ S_OP & S_O \end{bmatrix}$	(4.2)

From (3.17), it follows in each case that if the product $\bar{\sigma}[S_OP]\ \bar{\sigma}[P^{-1}T_O]$ is sufficiently large, then the approximation (3.13) to the structured singular value satisfies

$$\mu_2^{app}[M] \approx \sqrt{\bar{\sigma}[S_OP]\bar{\sigma}[P^{-1}T_O]} \quad . \tag{4.3}$$

Recalling our discussion in the previous section, large values of (4.3) indicate a potential lack of robustness with respect to small simultaneous perturbations. As has previously been recognized (see the references in [4]), plants whose transfer functions have a large condition number over some frequency range may yield feedback systems especially susceptible to the effects of simultaneous uncertainty. To show this using the present results, consider the following easily-derived bound, where $\kappa[P] \triangleq \bar{\sigma}[P]/\underline{\sigma}[P]$ denotes the condition number of the plant:

Lemma 4.1: At each frequency for which P^{-1} exists,

$$\underline{\sigma}[S_O]\ \underline{\sigma}[T_O]\ \kappa[P] \le \bar{\sigma}[S_OP]\bar{\sigma}[P^{-1}T_O] \le \bar{\sigma}[S_O]\ \bar{\sigma}[T_O]\kappa[P] \quad . \tag{4.4}$$

■

This bound shows that if $\kappa[P] >> 1$, then it is *necessary* that the product $\underline{\sigma}[S_O]\underline{\sigma}[T_O]$ be very small to prevent $\mu_2^{app}[M]$ from becoming large. A condition *sufficient* to prevent this difficulty is that the product $\bar{\sigma}[S_O]\bar{\sigma}[T_O]$ be very small. However, as the identity $S_O + T_O \equiv I$ reveals, $\bar{\sigma}[S_O]$ *and* $\bar{\sigma}[T_O]$ cannot both be small simultaneously. Furthermore, $\bar{\sigma}[S_O]$ is typically required to be small at low frequencies, while $\bar{\sigma}[T_O]$ is required to be small at high frequencies (if for no other reason than the fact that L_O must be strictly proper). Hence there necessarily exists a frequency range in which the product $\bar{\sigma}[S_O]\bar{\sigma}[T_O]$ cannot be small. If $\kappa[P] >> 1$ in that frequency range, then the upper bound in (4.4) will be large and potential robustness difficulties will exist.

Consider next the necessary condition $\underline{\sigma}[S_O]\underline{\sigma}[T_O] << 1$. Since

$$\underline{\sigma}[S_O]\underline{\sigma}[T_O] \geq \underline{\sigma}[S_O] \cdot |1 - \bar{\sigma}[S_O]| \quad , \tag{4.5}$$

it follows that the conditions $\underline{\sigma}[S_O] << 1$ and/or $\bar{\sigma}[S_O] \approx 1$ must be satisfied at frequencies for which $\kappa[P] >> 1$. Typically, the former condition will hold over a lower frequency range, while the latter will hold at sufficiently high frequencies. Note that there *necessarily* exists an overlapping frequency interval in which *both* these conditions must hold.

To illustrate, suppose that the plant has no poles or zeros in the right half plane, and consider the compensator

$$F(s) = g(s)P^{-1}(s) \quad , \tag{4.6}$$

where $g(s)$ is a scalar transfer function. In this case $T_I(s) = T_O(s) = g(s)/[1 + g(s)] \cdot I$ and $S_I(s) = S_O(s) = 1/[1 + g(s)] \cdot I$. It is straightforward to use classical design methods to choose $g(s)$ so that nominal design goals are satisfied and robustness with respect to individual perturbations is assured.[4] The fact that this compensation scheme allows a multivariable design problem to be reduced to a scalar problem is undoubtedly one reason for its frequent appearance in the literature; however, the ease of the scalar problem may be dangerously misleading when simultaneous uncertainty is present. To see this, note that

$$\bar{\sigma}[S_O P] \cdot \bar{\sigma}[P^{-1} T_O] = |g/(1+g)^2| \cdot \kappa[P] \quad . \tag{4.7}$$

so that the approximation (4.3) to the structured singular value is

$$\mu_2^{app}[M] \approx \sqrt{|g/(1+g)^2| \cdot \kappa[P]} \quad . \tag{4.8}$$

Typically, the coefficient $|g/(1+g)^2|$ will be small at low frequencies due to large open loop gain and small at high frequencies due to small open loop gain. At intermediate frequencies, where loop gain is approximately one, this coefficient *cannot* be small and thus, if $\kappa[P] >> 1$ at those frequencies, robustness against simultaneous perturbations will be poor. The approximation (4.3) and the actual value of $\mu[M]$ are illustrated in Figures 3 and 4 for the plant and compensator

$$P(s) = \begin{bmatrix} \dfrac{1}{s+1} & 0 \\ 0 & \dfrac{1}{s+2} \end{bmatrix} \cdot \begin{bmatrix} 9 & -10 \\ -8 & 9 \end{bmatrix}, \quad F(s) = [1/(.0159s)] \cdot P^{-1}(s) \tag{4.9}$$

Note that the agreement between these two plots is quite good at low-to-intermediate frequencies. In particular, the approximation predicts the existence of the peak in $\mu[M]$ near open loop crossover frequency $\omega = 63$ rad/sec. At high frequencies, the approximation (4.3) becomes less accurate. This is to be expected, since (3.19) shows that $\mu_2^{app}[M] \rightarrow \bar{\sigma}[S_0]$ at high frequencies.

For a discussion of alternate design methods, including some which are useful in maintaining $\bar{\sigma}[S_O P]\,\bar{\sigma}[P^{-1} T_O] << 1$, see[4].

We have shown that a spread in the singular values of S_O and T_O must exist in order to maintain stability robustness with respect to simultaneous perturbations. This property may conflict with other design goals. For example, suppose we require $\underline{\sigma}[T_O]$ to decrease at a lower frequency than does $\bar{\sigma}[T_O]$. This implies that the benefits of feedback associated with requiring $\bar{\sigma}[S_O]$ to be small cannot be maintained over as large a frequency range as would be possible otherwise. Indeed, it seems that the presence of simultaneous uncertainty imposes much different design goals than would be posed if individual uncertainties only were considered.

---

[4] Of course, the design must still satisfy the usual scalar design tradeoffs and limitations.

SECTION 5: CONCLUSIONS

The purpose of this paper has been to obtain insight into the robustness of a feedback system with respect to simultaneous uncertainty. We have done this by deriving an approximation to the structured singular value. The approximation appears useful in that it shows how the size of $\mu[M]$ is related to the size of the off-diagonal blocks of M.

It is straightforward to extend these results to the case of more than two blocks of uncertainty. To illustrate, suppose that $\Delta = diag\ [\Delta_1, \Delta_2, \Delta_3]$ and define $\overline{\Delta}_1 = diag\ [\Delta_1, \Delta_2]$. Partition M as $M = \begin{bmatrix} \overline{M}_{11} \overline{M}_{13} \\ \overline{M}_{31} M_{33} \end{bmatrix}$, and suppose that $\det[I + \overline{M}_{11}\, \overline{\Delta}_1] \neq 0$. Then Lemma 3.2 holds with Δ_1 replaced by $\overline{\Delta}_1$ and M_{22}, M_{21}, and M_{12} replaced by M_{33}, $\overline{M}_{31}$, and $\overline{M}_{13}$. Define the partitions $\overline{M}_{13} = \begin{bmatrix} M_{13} \\ M_{23} \end{bmatrix}$, and $\overline{M}_{31} = \begin{bmatrix} M_{31}\ M_{32} \end{bmatrix}$. It is then straightforward to show that the matrix $\widetilde{M}_{33}^{app}\ (\Delta_1)$, defined analogously to (3.7), is given by

$$\widetilde{M}_{33}^{app}(\Delta_1) = M_{33} - M_{31}\, \Delta_1\, M_{13} - M_{32}\, \Delta_2\, M_{23} \quad .$$

This expression may be used to derive approximations analogous to that in Corollary 3.4. We see that it is important to maintain *each* product $\bar{\sigma}\,[M_{3i}]\ \bar{\sigma}\,[M_{i3}]$ sufficiently small to prevent undue sensitivity to the effects of simultaneous perturbations.

The results of this paper need extension in two significant ways. First, it is desirable to have a *lower bound* on $\mu[M]$ which reflects the role of, say, the plant condition number in causing robustness difficulties. Such a lower bound may be obtained and will be discussed in a future report. Second, our analysis is concerned only with system properties at a *single* frequency. In fact, realizability considerations dictate that system properties in different frequency ranges cannot be manipulated independently. Incorporating such constraints into an analysis of simultaneous uncertainty appears to be a much more difficult task.

REFERENCES

[1] J.S. Freudenberg, D.P. Looze, and J.B. Cruz, Jr., "Robustness Analysis Using Singular Value Sensitivities," *Int. J. Control*, vol. 35, no. 1, p. 95, 1982.

[2] J.C. Doyle, "Analysis of Feedback Systems with Structured Uncertainties," *IEE Proc.*, vol. 129, Pt.D, no. 6, pp. 1-10, Nov. 1982.

[3] J.C. Doyle, J.E. Wall, Jr., and G. Stein, "Performance and Robustness Analysis for Structured Uncertainty," in *Proc. 21st IEEE Conf. on Decision and Control,* pp. 629-636, Orlando, FL, Dec. 1982.

[4] J.S. Freudenberg and D.P. Looze, "Relations between Properties of Multivariable Feedback Systems as Different Loop-Breaking Points: Part II," *Proc. 1986 American Control Conference,* Seattle, WA, June 1986.

[5] J.C. Doyle and G. Stein, "Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis," *IEEE Transactions on Automatic Control,* vol. AC-26, Feb. 1981.

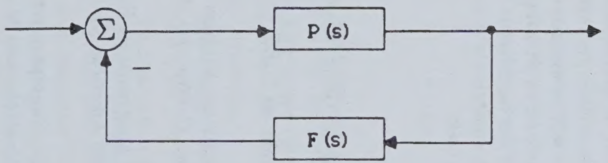


Figure 1: Block Diagram of Feedback System

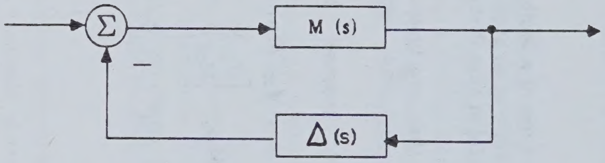


Figure 2: Rearranged Block Diagram for Structured Singular Value Analysis

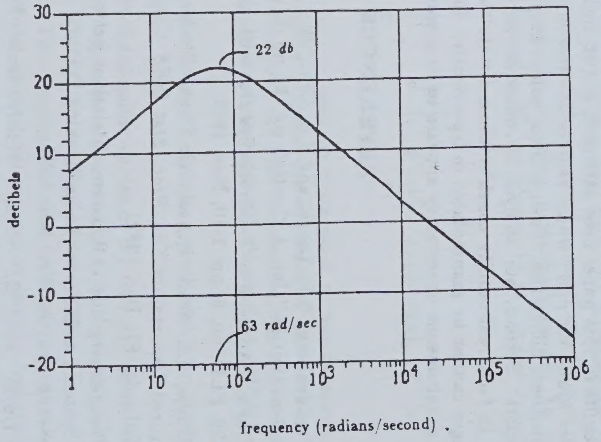


Figure 3: Approximation $\mu_2^{app}[M] \approx \sqrt{\kappa[P]\bar{\sigma}[S_O]\bar{\sigma}[T_O]}$

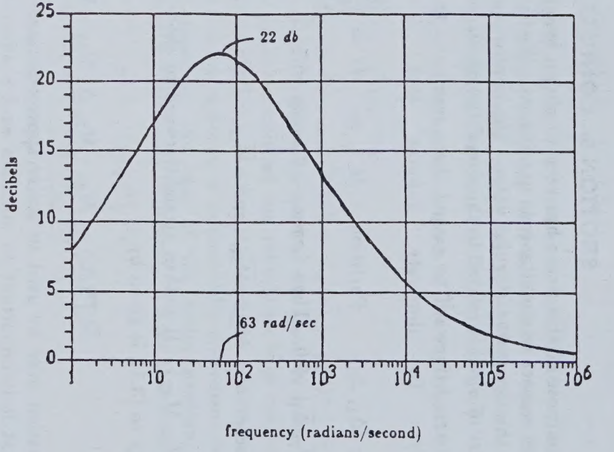


Figure 4: The Structured Singular Value, $\mu[M]$

A CAD Method and Associated Architectures for Linear Controllers

S. P. BOYD [1], V. BALAKRISHNAN, C. H. BARRATT,
N. M. KHRAISHI, X. LI, D. G. MEYER, S. A. NORMAN
Information Systems Laboratory
Department of Electrical Engineering
Stanford University
Stanford, CA 94305

Abstract— **New architectures and an associated CAD method are proposed for linear controllers. The architecture is suggested by recent results which parametrize all controllers that stabilize a given plant.**

With this architecture, the design of controllers is a convex programming problem which can be solved numerically even for large systems. Constraints on the closed-loop system such as asymptotic tracking, decoupling, limits on peak excursions of variables, step response settling time and overshoot, as well as frequency domain inequalities of any H_∞ variety are readily incorporated in the design. The minimization objective is general and can include LQG, H_∞, new ℓ_1 types, or any combination thereof.

To specify the constraints and objectives mentioned above, we are developing a *control specification language.* This control specification language will be the input to a *compiler* which will translate the specifications into a standard convex program in $\mathbf{R}^L$. A small but powerful subset of the language has been specified and its associated compiler implemented. The results are very encouraging.

The new architecture not only makes *design* of the controller simple but also its *implementation.* These controllers can be built right now from off the shelf components or integrated into silicon using standard VLSI cells.

Details will appear soon in a full length paper.

I. BASIC SETUP

The basic plant we consider is shown in Figure 1. We decompose its input into two signal vectors, w, the *exogenous inputs*, and u, the *control* or *actuator inputs.* Its outputs we decompose into y, the *measured* or *sensor outputs*, and z, the *regulated variables.* Our job is to design a controller K, with input y *only*, and output u.

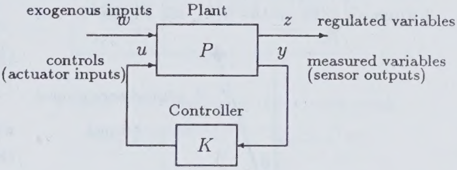


Figure 1: Basic plant and controller.

The signal y represents the signal actually accessible to the controller K, including any command inputs, which may be considered exogenous inputs (i.e. components of w) passed directly to some of components of y.[2] Similarly the signal u represents those inputs to the plant which our controller may vary, that is, those inputs to the plant manipulable by the controller. Thus it is by *definition* that the controller has input y only and output u only.

The exogenous input, w, will include real physical disturbances or noises (torques, forces) acting on the plant, any actuator or sensor noise, and, as mentioned above, any

---

[1] Research supported in part by ONR under N00014-86-K-0112 and NSF under ECS-85-52465.

[2] We may think of the exogenous inputs which represent commands as the actual physical commands inputs, e.g. angle of a potentiometer, and the corresponding components of y as the 'sensed command', e.g. voltage output from a potentiometer.

command inputs. w may also contain fictitious inputs injected anywhere in the plant.

The signal z represents any system signal about which we will express a specification, regardless of its accessibility to the controller. Obvious examples are the actual positions, forces, temperatures, etc. we wish to regulate or control. z may include 'internal states' or variables we wish to limit. Some components can be fictitious, e.g. linear combinations of state variables or even filtered versions of signals. z may also include components of u or y.

Thus H_{zw}, the multi-input multi-output closed-loop map from w to z, contains, by definition, every *closed-loop* map of interest.

To illustrate this decomposition of inputs and outputs, we present a simple example. A motor with shaft encoder is used to regulate the angle, θ, of an arm to some (small) commanded angle, θ_{ref}, while a disturbance torque, d, acts on the arm; see Figure 2. Let θ_{se} denote the shaft encoder output, and let $n_{sens} = \theta - \theta_{se}$, the difference between the actual angle and the sensed angle, that is, sensor quantization 'noise'. Exogenous inputs are

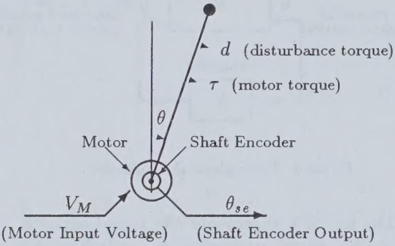


Figure 2: Example system.

$$w = \begin{bmatrix} \theta_{ref} \\ d \\ n_{sens} \end{bmatrix}.$$

There are two sensed outputs,

$$y = \begin{bmatrix} \theta_{ref} \\ \theta_{se} \end{bmatrix}.$$

Note that the command θ_{ref} is simply passed through P; thus the block diagram of Figure 1 could be drawn in a more conventional way as a two degree of freedom SISO controller.

There is only one actuator input, the motor input voltage, V_m; that is, $u = V_m$. A possible choice of regulated variables is

$$z = \begin{bmatrix} \theta \\ \theta - \theta_{ref} \\ \tau \\ V_m \end{bmatrix}$$

where τ is the motor torque. We will refer to this example several times in the sequel.

We assume only that the I/O map of the plant, P, is linear. For the moment, we do not say whether the system is continuous or discrete time, since much of the ensuing discussion is independent of this. In any case, a real system is almost certainly hybrid with a continuous time physical plant and a discrete, possibly multirate, controller.

We partition the map P as

$$P = \begin{bmatrix} P_{zw} & P_{zu} \\ P_{yw} & P_{yu} \end{bmatrix}$$

so that

$$\begin{bmatrix} z \\ y \end{bmatrix} = P \begin{bmatrix} w \\ u \end{bmatrix}.$$

Of course we have

$$H_{zw} = P_{zw} + P_{zu}K(I - P_{yu}K)^{-1}P_{yw} \quad (1)$$

where H_{zw} is the closed-loop map from the exogenous inputs w to the regulated variables z.

Note that H_{zw} depends on the controller K in a linear fractional fashion. A relatively simple constraint on H_{zw}, e.g. that a certain entry be zero, corresponds via (1) to a much more complicated constraint on the controller K. We remark here that if $P_{yu} = 0$ then (1) simplifies considerably to

$$H_{zw} = P_{zw} + P_{zu}KP_{yw}.$$

Here H_{zw} is *affine* in K. Note that this corresponds to the case where there is essentially *no feedback* in the system.

II. PARAMETERIZATION OF STABILIZING CONTROLLERS

In this section we discuss the recent parametrization of all stabilizing controllers. Precise definitions and all details can be found in Vidyasagar's book [1].

A basic requirement is that the controller stabilize the plant. A recent major theme in control theory is that the set of H_{zw}'s achievable with controllers which stabilize the system is a *linear variety*, that is, a translation of a linear subspace.

A linear variety, $\mathcal{L}$, in a vector space, $\mathcal{V}$, is often described as the nullspace or range of an affine map from or to $\mathcal{V}$. Thus we might have

$$\mathcal{L} = \{v \in V \mid Av = b\}$$

where A is a linear map from $\mathcal{V}$ to $\mathcal{W}$ and $b \in \mathcal{W}$. This is a description of $\mathcal{L}$ in terms of a *linear equality constraint*; if $\mathcal{W}$ is $\mathbf{R}^k$, then we can interpret each entry of $Av = b$ as a single linear functional equality constraint in $\mathcal{V}$. Let $\mathcal{H} = \{H_{zw} \mid \text{system is closed-loop stable}\}$, the set of achievable H_{zw}'s. $\mathcal{H}$ may be described via linear equality constraints using the interpolation conditions [3]

Alternatively we may describe a linear variety, $\mathcal{L}$, as the range of an affine map:

$$\mathcal{L} = \{Cu + d \mid u \in \mathcal{U}\}$$

where C is a linear map from $\mathcal{U}$ to $\mathcal{V}$ and $d \in \mathcal{U}$. Such a representation can be regarded as a *free parametric representation* of $\mathcal{L}$; u is a free parameter.

A free parametric representation of $\mathcal{H}$ can be derived using stable coprime fractional theory [3,1] One standard form is the 'Q-parametrization':

$$\mathcal{H} = \{T_1 + T_2 Q T_3 \mid Q \text{ stable}\}.$$

where T_1, T_2, and T_3 are some stable maps.

Note that the Q-parametrization

$$H_{zw} = T_1 + T_2 Q T_3 \tag{2}$$

has exactly the same form as the no-feedback formula mentioned in Section 1; we will soon see a block diagram interpretation of the Q-parametrization in which the parameter Q 'sees' no feedback. In particular, H_{zw} is affine in Q.

The derivation of the Q-parametrization starts with any controller, K_o, which stabilizes the plant and which we will call the *nominal controller*. Let $K_o = Y^{-1}X$ be a left stable coprime factorization [1] of the nominal controller and $P_{yu} = \tilde{D}^{-1}\tilde{N}$ be a left stable coprime factorization of P_{yu}. Then every controller of the form

$$K = (Y - Q\tilde{N})^{-1}(X + Q\tilde{D}), \; Q \text{ stable}$$

stabilizes P and conversely every controller which stabilizes P has this form for some stable Q. There is a similar characterization of the stabilizing controllers in terms of right coprime factorizations.

Since K_0 stabilizes P_{yu}, there is a right coprime factorization of P_{yu}. with $XN + YD = I$ = the identity map. A little calculation yields the Q-parametrization formula, $H_{zw} = T_1 + T_2 Q T_3$, with

$$T_1 = P_{zw} + P_{zu} D X P_{yw},$$

$$T_2 = P_{zu} D,$$

$$T_3 = \tilde{D} P_{yw}.$$

A simple block diagram of the Q-parametrization is shown in Figures 3 and 4.

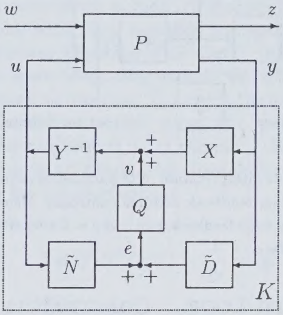


Figure 3: Block diagram of Q-parametrization.

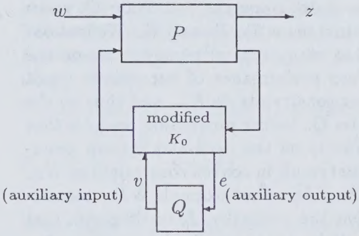


Figure 4: Q-parametrization as modification to nominal controller.

T_1 is simply the H_{zw} achieved with the nominal controller K_o; T_2 is the map from w to e, T_3 is the map from v to z (see Figure 4). The key to the parametrization is that the closed-loop map from v to e is *zero*, so that Q 'sees' no feedback.

Doyle [4] has given a very nice interpretation of the Q-parametrization when the nominal controller is an estimated state feedback. In this case e is simply the output prediction error, $\hat{y} - y$, of the observer, and v is just added before the observer tap to the output of the nominal controller as shown in Figure 5. The controller shown in Figure 5 is sometimes called an *observer-based controller* (OBC); the point is that *every* controller which stabilizes P can be realized as an observer-based controller.

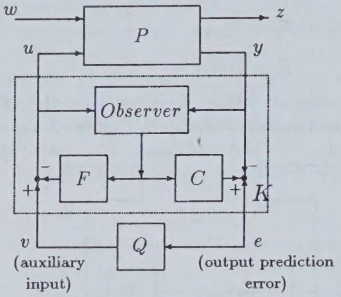


Figure 5: Doyle's interpretation of Q-parametrization for estimated state feddbcak nominal controller. Here F is a stabilizing state feedback gain and $y = Cx$ where x is the plant state.

III. CLOSED-LOOP CONSTRAINTS AND OBJECTIVES

In this section and the next, we consider the problem of designing the parameter Q, given the nominal maps T_1, T_2, and T_3. We first observe that many typical requirements on the *closed-loop* performance of our system result in *convex* constraints on H_{zw} and thus on the parameter Q. Before proceeding, we note that requirements on the *open-loop* system generally do *not* result in convex constraints on H_{zw} or Q. One important example is the requirement that the controller, K be diagonal, that is, that K be a decentralized controller. Another important example is the requirement that the controller be stable.

Suppose our design problem is specified as

$$\begin{array}{ll} \text{minimize} & \Phi(H_{zw}) \\ & H_{zw} \in \mathcal{K} \\ & \text{cl. loop stable} \end{array}$$

where Φ is a convex functional and $\mathcal{K}$ a convex (constraint) set. This is equivalent to

$$\begin{array}{ll} \text{minimize} & \tilde{\Phi}(Q) \\ & Q \in \tilde{\mathcal{K}} \end{array} \tag{3}$$

where

$$\tilde{\mathcal{K}} = \{Q \mid T_1 + T_2QT_3 \in \mathcal{K}\}$$

and

$$\tilde{\Phi}(Q) = \Phi(T_1 + T_2QT_3).$$

Note that $\tilde{\Phi}$ and $\tilde{\mathcal{K}}$ are convex. Of course $\tilde{\mathcal{K}}$ may be empty; this simply means that *no stabilizing controller* can satisfy the constraint $H_{zw} \in \mathcal{K}$.

We now list some typical constraints (and objectives) on H_{zw}, some collection (sum) of which might describe the constraint set $\mathcal{K}$ (objective functional Φ).

For purposes of discussion, we assume the system is discrete-time. H will denote the transfer matrix of H_{zw}, h its impulse response matrix, and $s(t) = \sum_{i=0}^{t} h(i)$ its step response. Of course a convex constraint or functional of H, h, or s is a convex constraint or functional of H_{zw}.

Where possible, we will refer to the example system of §1, which we presume has been sampled at some appropriate rate. When referring to this system, we will use *symbolic* subscripts written in square brackets so the reader need not refer to §1 to find which components of w and z have which interpretations. Thus we will write $H[\theta][\theta_{ref}]$ instead of H_{11}.

- *Asymptotic Tracking, Decoupling, and Regulation*

  The step response from some command input to the regulated variable it is supposed to control must converge to one, e.g.

  $$\lim_{t\to\infty} s[\theta][\theta_{ref}](t) = 1.$$

  Equivalently,

  $$H[\theta][\theta_{ref}](e^{j0}) = 1.$$

  This constraint is a single linear functional equality constraint on H, hence, of course, a convex constraint.

  Asymptotic tracking of ramps or more complicated inputs can be handled as two or more linear functional equality constraints, e.g. $H[\theta][\theta_{ref}](e^{j0}) = 1$, $H'[\theta][\theta_{ref}](e^{j0}) = 0$.

  Asymptotic regulation and *asympototic decoupling* are similar constraints. We may require that a regulated variable asymptotically reject constant inputs appearing at certain exogeneous inputs. When the exogenous input is another command input, this constraint is asymptotic decoupling; when the exogenous input is some disturbance, this constraint is asymptotic regulation. For example, to ensure that θ asymptotically rejects any constant disturbance torque, we might specify

$$H[\theta - \theta_{ref}][d](e^{j0}) = 0.$$

- *Overshoot, Undershoot, and Settling-Time Limits*

  We may require that some step response lie in the unshaded region of Figure 6, e.g.

  $$-0.3 \leq s[\theta][\theta_{ref}](t) \leq 1.3; \qquad 0 \leq t \leq 10,$$
  $$|s[\theta][\theta_{ref}] - 1(t)| \leq 1/(t+1); \qquad t > 10.$$

  This constraint can be expressed as a collection of linear functional ineqqqualities on s (hence H_{zw}): $L(t) \leq s_{32} \leq U(t)$ for $t = 0, 1, ...$ Thus the set of H_{zw}'s satisfying this constraint is *convex* (it may of course be empty).

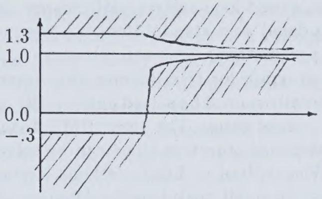


Figure 6: Step response overshoot, undershoot, and settling time constraints.

- *Bounds on Closed-Loop Signal Peaks*

  Given bounds, W_j, on each exogenous input, we may require that each regulated variable be bounded by some given maximum Z_i. This constraint could arise from the requirement not to saturate an actuator or sensor or exceed some internal variable force, torque, or current limit, and is expressed as

  $$\sum_{j=1}^{N_{exog}} W_j \sum_{t=0}^{\infty} |h_{ij}(t)| \leq Z_i \text{ for } i = 1, ...N_{reg}$$

  where N_{exog} is the number of exogenous inputs and N_{reg} the number of regulated variables.

  Thus the constraint

  $$0.1 \sum_{t=0}^{\infty} |h[V_m][\theta_{ref}](t)|$$
  $$+ 0.03 \sum_{t=0}^{\infty} |h[V_m][d](t)|$$
  $$+ 0.001 \sum_{t=0}^{\infty} |h[V_m][d](t)| \leq 50$$

  will guarantee that whenever the command input θ_{ref} is bounded by 0.1, the disturbance d by 0.03, and the sensor noise n_{sens} by 0.001, the motor voltage input V_m will be bounded by 50.

  Again, this is a convex constraint on h, and thus on H_{zw}.

- *Small RMS Disturbance Response*

  If the exogenous input, w, is driven by a wide-sense stationary stochastic process with some specified spectral distribution, then $\mathrm{E}z(t)^T G^T G z(t)$, where G is some weighting matrix, should be as small as possible. Typically, the stochastic process would have zero power in the exogeneous input channels which are command inputs, and G would have only a few nonzero entries.

  We have

  $$\mathrm{E}z(t)^T G^T G z(t) = \frac{1}{2\pi} \times \text{ the Trace of:}$$

  $$G\left(\int_0^{2\pi} H(e^{j\Omega}) S_w(e^{j\Omega}) H(e^{-j\Omega})^T d\Omega\right) G^T$$

  where S_w is the power spectral density of the stochastic process driving w. This objective is a *nonnegative quadratic functional* of H and thus of H_{zw}, in particular it is convex.[3]

  For our example, a possible S_w would be

  $$S_w(e^{j\Omega}) = \begin{array}{c} \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & S_d(e^{j\Omega}) & 0 \\ 0 & 0 & \Delta^2/12 \end{array}\right] \end{array} \begin{array}{l} (\theta_{ref}) \\ (d) \\ (n_{sens}) \end{array}$$

  where $S_d(e^{j\Omega})$ is the spectral density of the disturbance torque and Δ is the step size of the shaft encoder. Note that we put no noise power in the command input θ_{ref}. If we are interested in the noise power in the pointer angle, θ, then we let $G = [\,1\;0\;0\;0\,]$ and the functional above is simply

  $$\mathrm{E}\theta^2 = \frac{1}{2\pi} \times$$
  $$\left[\int_0^{2\pi} |H[\theta][d](e^{j\Omega})|^2 S_d(e^{j\Omega})\, d\Omega\right.$$
  $$\left. + \int_0^{2\pi} |H[\theta][n_{sens}](e^{j\Omega})|^2 \Delta^2/12\, d\Omega\right]$$

---

[3] Here *quadratic functional* includes a *linear functional* and a *constant (functional)* term.

which can be interpreted as the total noise power in θ due to the disturbance and sensor noise.

- *Bounds on Transfer Function Peak Magnitudes*

  We may specify an upper bound for some entry (or block of entries) of H, e.g. $|H_{65}(e^{j\Omega})| \leq U(\Omega)$ for all Ω, where $U(\Omega)$ is some bound function. This constraint may arise in several ways.

  First, many classical specifications of bandwidth or peaking are expressed this way. Referring to our example, we might specify

  $$|H[\theta-\theta_{ref}][\theta_{ref}](e^{j\Omega})| \leq 0.03 \text{ for } |\Omega| \leq \Omega_B$$

  so that for frequencies below Ω_B, we get -30db of rejection.

  $$|H[\theta-\theta_{ref}][\theta_{ref}](e^{j\Omega})| \leq 3 \text{ for all } \Omega$$

  to keep gain in the command to tracking error map under about 10db for all frequencies.

  The second way a transfer function peak magnitude constraint can arise is as follows. If we require a small RMS response of some regulated variable, as in the previous constraint, but know only a bound on the total power in the process driving the exogenous input w, and not its spectral distribution, then we are led to a transfer function peak magnitude constraint; this is the basis of H_∞ control theory.

  Third, a *sufficient* condition to stabilize not just the plant, but all plants 'near' our given plant (in a peak frequency response deviation sense), may be formulated as a bound on certain closed-loop frequency response magnitudes.

  These H_∞ type constraints are convex constraints on H and hence H_{zw}.

- *Miscellaneous Bounds and Objectives*

  We mention here some less common constraints. A *slew-rate* constraint on the step response, say s_{11}, may be enforced as:

  $$|h_{11}(t)| \leq R \text{ for all } t.$$

  A *passivity* or *minimum dissipation* constraint, say on the 1, 1 entry of H_{zw}, can be formulated as

  $$\Re H_{11}(e^{j\Omega}) \geq -D \text{ for all } \Omega.$$

  Such a constraint can be used, for example, to guarantee stability of the system with a saturating actuator. All constraints involving step responses can be generalized to other, arbitrary, input signals, or even sets of signals. Thus we may specify a maximum peak tracking error for command inputs bounded by B_{cmd} and slew rate under R_{cmd}.

Finally we mention that the boundary between constraint and objective is not sharp; we could, for example, try to minimize an overshoot in a certain step response, or put hard constraints on the RMS response.

At this point it should be clear that a large number of practical constraints on, and objectives for, closed-loop system performance can be formulated as a standard convex program for the parameter Q.

This program for Q is infinite dimensional and generally cannot be solved analytically except in special cases. The small RMS disturbance response objective alone can be solved using Wiener-Hopf or LQG methods; certain forms of the small peak transfer function objective problem are solved by H_∞-optimal control theory; certain forms of the small peak disturbance objective problem are solved by the new l_1-optimal control theory introduced by Vidyasagar [5] and developed by Dahleh and Pearson [2].

IV. SPECIFICATIONS TO CONVEX PROGRAM COMPILER

The translation from closed-loop constraints or objective on H_{zw} to convex program for the parameter Q is mathematically straightforward. As mentioned above, the resulting program is infinite dimensional. An approximate solution can be found numerically when the parameter Q is restricted to a large, but finite, dimensional space. Let

$$Q = \sum_{i=1}^{L} x_i \Gamma_i$$

where $x_i \in \mathbf{R}$ and Γ_i are fixed stable maps. We will call $x = [x_1, ..., x_L]^T$ the *decision variables*. With this additional restriction, we have a standard finite dimensional convex program in $\mathbf{R}^L$

$$\begin{array}{cc} \text{minimize} & f(x) \\ x \in \hat{\mathcal{K}} & \end{array} . \qquad (4)$$

While the convex constraint set $\hat{\mathcal{K}}$ cannot be explicitly found, it is easily approximated, and then this explicit finite dimensional convex program can be solved numerically by any of a number of methods.

Let us first briefly discuss the choice of parametrization and the Γ_i's, since this affects how well the finite dimensional program (4) approximates the program (3).

In theory, it makes no difference which nominal controller nor which coprime factorization is used to form the T_i's in the Q-parametrization; all parametrizations produce precisely the same set, $\mathcal{H}$, of H_{zw}'s. The *particular* Q which yields a given H_{zw}, however, will depend on the choice of T_i's. As mentioned above, the Q's will be restricted to some finite dimensional space, so the set of H_{zw}'s achievable *in practice* is a finite dimensional linear variety containing T_1, which is the H_{zw} achieved with the nominal controller. This means that the nominal controller used should yield at least reasonable performance, so that we have more of a chance of finding a 'moderate' Q which achieves good performance. By 'moderate' we mean a Q which is neither too 'big' nor too 'stiff', that is, has too broad a spread of dynamics. An inappropriate nominal stabilizing controller will place severe demands on the filter Q. For example, a state feedback which places the eigenvalues far to the left of the bandwidth achievable when other factors such as control authority are taken into account, will require a 'large' Q filter, essentially to cancel the unreasonably large plant input signal generated by the excessive state feedback. So in practice, the choice of nominal controller is not arbitrary.

A similar comment applies to the choice of coprime factorizations, which will affect the maps T_2 and T_3. Roughly speaking, they should be of reasonable size and have most of their dynamics not too far from the final closed-loop bandwidth.

We turn now to the practical problem of translating a large number of constraints/objective on step responses, transfer functions, RMS responses, and so on, into the corresponding convex program for the decision variables. We propose that this job be mechanized by the use of a *compiler* which accepts as input a list of closed-loop constraints and a description of the objective and as output produces the convex program for the decision variables. The constraints and objective would be specified in a *control specification language*, which would be natural for the control engineer, refering directly to step responses, noise powers, transfer functions, etc. The output of the compiler would be a convex program which could be solved by a convex program solver, as shown in Figure 7.

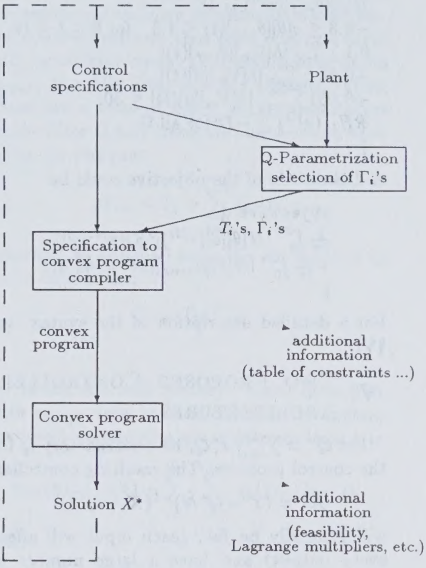

Control specifications
Plant
Q-Parametrization selection of Γ_i's
T_i's, Γ_i's
Specification to convex program compiler
convex program
additional information (table of constraints ...)
Convex program solver
Solution X^*
additional information (feasibility, Lagrange multipliers, etc.)

Figure 7: Design of approximately optimal controller using control specification to convex program compiler.

IV.A Control Specification Language

Here we discuss the *control specification,* the input to the compiler. Many of the features we discuss have actually been implemented in our compiler; §VI describes what we have implemented so far.

An overall general structure for the control specification is

```
constraints {
   <constraint>;
   ...
   <constraint>;
}
```

```
objective {
   description of objective
}
```

For example, the constraints might have the forms encountered in §III, such as

`constraints {`

$$\begin{array}{l} H[\theta][\theta_{ref}](e^{j0}) = 1; \\ -0.3 \leq s[\theta][\theta_{ref}](t) \leq 1.3 \quad \text{for } 0 \leq t \leq 10; \\ 0.1 \sum_{t=0}^{\infty} |h[V_m][\theta_{ref}](t)| \\ +0.03 \sum_{t=0}^{\infty} |h[V_m][d](t)| \\ +0.001 \sum_{t=0}^{\infty} |h[V_m][d](t)| \leq 50; \\ \Re H_{11}(e^{j\Omega}) \geq -D \text{ for all } \Omega; \end{array}$$

`}`

A description of the objective could be

`objective {`

$$\begin{array}{l} \frac{1}{2\pi} \int_0^{2\pi} |H[\theta][d](e^{j\Omega})|^2 S_d(e^{j\Omega})\, d\Omega \\ +\frac{1}{2\pi} \int_0^{2\pi} |H[\theta][n_{sens}](e^{j\Omega})|^2 \frac{\Delta^2}{12}\, d\Omega \end{array}$$

`}`

For a detailed description of the syntax, see §VI.

V. TWO PROPOSED CONTROLLER ARCHITECTURES

Let $Q^* = \sum_{i=1}^{L} x_i^* Q_i$ be a satisfactory Q for the control problem. The resulting controller

$$K^* = (Y - Q^* \tilde{N})^{-1}(X + Q^* \tilde{D})$$

will generally be *full*, (each input will affect every output) and have a large number of states, perhaps hundreds. The standard procedure at this point would be to apply some sort of model reduction to arrive at a low order controller, K_{red}, with only tens of states. We propose instead two methods for implementing the full K^*, both based on the Q-parametrization. Both methods rely on the availability and ease of VLSI implementation of many-tap FIR filters, used extensively in signal processing, for example in equalizers for modems. Thus, our controllers will tend to have far greater orders than is currently common. While it is certain that there is a much lower order K_{red} which provides satisfactory performance, we question the need to find it.

V.A MODIFIED NOMINAL CONTROLLER ARCHITECTURE

Consider the block diagram of the Q-parametrization given in Figure 4. We propose that controllers *actually be built with the architecture shown in Figure 4*, with Q realized (possibly) separately in hardware as an FIR filter. This architecture can be thought of as a simple modification of the nominal controller; the nominal controller must be modified to yield an auxiliary output signal e and accept an auxiliary input signal v. In some controllers, the signal e is easily available, even though Figure 3 suggests that we need an additional $\tilde{N}$ and $\tilde{D}$. Similarly, injecting the signal v as shown in Figure 4 is also straightforward.

For example, if the nominal controller is an estimated state feedback, then as mentioned earlier, e is the output prediction error of the observer, and v is just added to the output of the nominal controller, before the observer tap, as shown in Figure 5. Thus, in many cases the hardware and/or computational costs of modifying the nominal controller are slight, since the Q filter does all of the additional processing.

We note that this method can be used to improve *existing* controllers.

V.B FIR COPRIME ARCHITECTURE

Generally, the nominal controllers which provide easy access to auxiliary output e are those with complexity on the order of the plant, the estimated state feedback nominal controller a good example. The reason is, roughly speaking, that e is the pseudo-state observer error, which means that the modified nominal controller really contains a pseudo-state observer within it, which is likely to have the same complexity as the plant. This is admittedly vague, and only offered as our interpretation.

In nominal controllers of much lower complexity than the plant, e.g., a diagonal PI controller, e is usually not easily accessible, and must be reconstructed. In this case we propose that the entire controller be implemented using FIR filters.

Consider the coprime factorization of the controller K^* given above. Suppose Y, Q^*, $\tilde{N}$, X, and $\tilde{D}$ are all FIR, or very nearly approximated by FIR filters. Recall that unlike the plant or controller, these operators are *stable* and hence their impulse matrices decay to zero. With proper choice of sample rate and parametrization, 100 tap FIR filters should be more than adequate. In this case, K^* can be realized as a feedback connection around one FIR filter, as we now show. K^* is governed by the equations

$$Yu - Xy = v = Q\tilde{N}u + Q\tilde{D}$$

(see Figure 3). We rewrite these equations as

$$u = (I - V)u + Xy + v$$

$$v = Q\tilde{N}u + Q\tilde{D}y$$

which we interpret as an FIR filter, F, with input u, y and v, and output u and v, and

$$F = \begin{bmatrix} I - V & X & I \\ Q\tilde{N} & Q\tilde{D} & 0 \end{bmatrix}.$$

This is shown in Figure 8. Of course there are many other F's which realize K^*, and we do not yet know a good procedure for picking one.

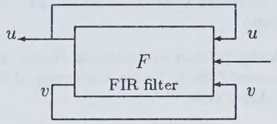


Figure 8: FIR coprime architecture for controllers.

Finally, we mention that for both architectures, it is possible to vary the effect of the parameter Q from 'off' to 'on' by inserting a gain, α, which varies between zero and one, in the right place (e.g. in the bottom loop of Figure 8). Thus we have a *homotopy* (continuous deformation) of our nominal controller K_0 into our designed controller K^*, with all intermediate controllers stabilizing the plant, indeed with H_{zw} affine in the parameter α. Such a parameter might be quite useful as a *control authority parameter*. If our nominal controller has somewhat lower authority/bandwidth than our designed controller, but is more robust to plant variations or failures, then α can vary the control authority and robustness of our controller from nominal (resp., low and high) to designed (resp., high and low). In an emergency, we can smoothly drop back to the nominal controller. Of course, this will only work with moderate plant variations, and cannot take the place of a real fault tolerant control system (with fault detection and reconfiguration).

VI. IMPLEMENTATION OF A SUBSET

We have implemented a compiler which recognizes a small but powerful subset of *control specification language*. Time and frequency domain linear and magnitude frequency inequality constraints, and quadratic objective are recognized. The system must have only one actuator, and the filter Q is simply a FIR filter. We are currently implementing a much more general compiler.

We assume Q has n_{iq} inputs and *a single output*; so far we have implemeted a compiler for control problems that have only one actuator or control input. Q is assumed to be an n_{coeff}-tap FIR filter in every channel. Thus, there are a total of $n_{iq} \times n_{coeff}$ parameters in the filter Q and these are the *decision variables*. In this case

$$H_{zw} = T_1 + \sum_{i=1}^{n_{iq}} Q_i G_i$$

where Q_i is the map from the ith input of Q to its output and

$$G_i = T_2 e_i^T T_3$$

(e_i is the *ith* unit vector). Since Q is FIR with n_{coeff} taps in each channel, the system impulse response at a given discrete time k is

$$h_{zw}(k) = t_1(k) + \sum_{i=1}^{n_{iq}} \sum_{j=0}^{n_{conv}} q_i(j) g_i(k - j),$$

where

$$n_{conv} \triangleq \min(n_{coeff} - 1, k).$$

Clearly the system impulse response at any discrete instant, k, is affine in the decision variables, $q_i(j)$.

Our compiler recognizes three time domain functionals:

$$ustep[i][j](k),$$
$$h[i][j](k),$$
$$q[i](k).$$

These are, respectively, the step and impulse responses from input j to output i at time k, and the step response of Q from it's input i at time k (recall that Q has a single output).

Three frequency domain functionals are recognized:

$$Re\_H[i][j](r,\theta),$$
$$Im\_H[i][j](r,\theta),$$
$$Mag\_H[i][j](r,\theta).$$

These are, respectively, the real part, imaginary part, and magnitude of the frequency response from input j to output i at $re^{j\theta}$, r and

θ are real. Since our current convex programming package allows only linear constraints, constraints involving $MagH$ are parsed into two contraints on the real and imaginary parts of H— at the expense of at most a 1.5dB error. It could be modified to parse into more constraints, yielding less error.

Real scalar variables and scalar expressions involving standard mathematical functions are allowed.

Functional inequalities are allowed to take the form

$$f \leq e;$$

$$f \geq e;$$

$$f = e;$$

$$e \leq f \ \leq \ e;$$

$$\mid f \mid \leq e;$$

where f represents one of the above functionals and e represents any scalar expression.

A quadratic objective can be specified as the sum of any number of the following terms:

$h\_sqr[i][j](k)$,
$Mag\_H\_sqr[i][j](r, \theta)$,
$q\_sqr[i](k)$.

These are, respectively, the square of the impulse response from input j to output i at time k, the square of the frequency response magnitude from input j to output i at $re^{j\theta}$, and the square of the impulse response of Q from its input i to it's output at time k.

Nested looping is allowed. This makes repetitive constraints and objective terms easy to specify, for example

```
constraints {
for $t = 0$ to 10
    $-0.3 \leq ustep[\theta][\theta_{ref}](t) \leq 1.3$;
}
objective {
for $\Omega = 0$ to $\pi$ step 0.03
    $0.03 * Mag\_H\_sqr[\theta][d](e^{j\Omega})$;
}
```

The standard C language preprocessor, **cpp**, is used, which allows the use of macros and file include facilities.

VII. ACKNOWLEDGEMENTS

Various preliminary forms of this material were presented at seminars at Berkeley, USC, Cal Tech, and Honeywell SRC; several people made substantial suggestions (which we took): C. A. Desoer, J. Doyle, M. Safonov, P. Kokotovic, and R. Kosut. Also, the authors wish to thank Bear M. Claw for numerous perceptive comments.

REFERENCES

[1] M. Vidyasagar, *Control System Synthesis: A Factorization Approach,* MIT Press, 1985

[2] M.A. Dahleh and J.B. Pearson, "ℓ_1 Optimal feedback controllers for discrete-time systems." Technical report. TR 8513. Rice University, Houston, TX 77251 Sept. 1985

[3] C.A. Desoer, R.W. Liu, J. Murray, and R. Saeks, "Feedback system design: The fractional represenation approach to analysis and synthesis." *IEEE Trans. Automat. Contr.*, vol AC-25, pp. 399-412, June. 1980

[4] J.C. Doyle, "Matrix interpolation theory and optimal control." Ph.D Thesis, University of California, Berkeley 1984

[5] M. Vidyasagar, "Optimal rejection of persistent bounded disturbances." *IEEE Trans. Automat. Contr.*, vol AC-31, pp. 527-535, June 1986

THE INADEQUACY OF INPUT-OUTPUT INFORMATION IN ROBUST COMPENSATOR DESIGN

J. DANIEL COBB
Department of Electrical and Computer Engineering
University of Wisconsin
Madison, WI 53706

Recent work by Vidyasagar [1], Khalil [2], Rohrs [3], and Ioannou and Kokotovic [4] has addressed the problems caused by unmodelled dynamics or parasitics in a closed-loop control system. We are specifically interested in the results of [1] where it is shown that, for any plant (A,B,C,D), any singularly perturbed augmentation of that plant

$$\begin{bmatrix} I & 0 \\ 0 & \varepsilon I \end{bmatrix} \dot{x} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} x + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u \qquad \text{(1a)}$$

$$y = \begin{bmatrix} C_1 & C_2 \end{bmatrix} x$$

where

$$A_{11} - A_{12}A_{22}^{-1}A_{21} = A, \; B_1 - A_{12}A_{22}^{-1}B_2 = B \qquad \text{(1b)}$$

$$C_1 - C_2A_{22}^{-1}A_{21} = C, \; -C_2A_{22}^{-1}B_2 = D$$

and any fixed strictly proper compensator, if the corresponding closed-loop system is stable at $\varepsilon = 0$, then so is the perturbed closed-loop system for sufficiently small $\varepsilon > 0$. We wish to explore the consequences of perturbing the compensator as well. Let (F,G,H,0) be any compensator and consider perturbations of the form (1).

Theorem 1 Let $R < \infty$ and $\gamma > 0$. There exist $\varepsilon_0 > 0$ and singular perturbations (1) of the plant and compensator such that the closed-loop configuration of the perturbed systems has a pole p_ε satisfying $|p_\varepsilon| > R$ and $|\arg p_\varepsilon| < \gamma$ whenever $0 < \varepsilon < \varepsilon_0$.

Theorem 1 says that perturbations of plant and compensator can always be found which destabilize the closed-loop system. The proof of the theorem is based on examination of the perturbed closed-loop system

$$\begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & \varepsilon I & 0 \\ 0 & 0 & 0 & \varepsilon I \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{z} \\ \dot{\xi} \\ \dot{\varsigma} \end{bmatrix} = \begin{bmatrix} A_{11} & B_1H_1 & A_{12} & B_1H_2 \\ G_1C_1 & F_{11} & G_1C_2 & F_{12} \\ A_{21} & B_2H_1 & A_{22} & B_2H_2 \\ G_2C_1 & F_{21} & G_2C_2 & F_{22} \end{bmatrix} \begin{bmatrix} x \\ z \\ \xi \\ \varsigma \end{bmatrix} + \begin{bmatrix} B_1 \\ 0 \\ B_2 \\ 0 \end{bmatrix} v$$

$$y = \begin{bmatrix} C_1 & 0 & C_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ z \\ \xi \\ \zeta \end{bmatrix}$$

It can be shown that A_{22}, B_2, C_2, F_{22}, G_2, and H_2 can always be found to make the matrix

$$\begin{bmatrix} A_{22} & B_2H_2 \\ G_2C_2 & F_{22} \end{bmatrix}$$

unstable; hence, the closed-loop system is unstable for $\varepsilon > 0$.

An important property of the perturbations (1) is that, in a certain sense, they exhibit the same input-output behavior as the nominal system. Specifically, let $\mathcal{U}$ = all C^1 functions $u:[0,\tau] \longrightarrow \mathbb{R}^m$ with $u(0) = 0$, $\max \|u(t)\| < K_0$, and $\max \|\dot{u}(t)\| < K_1$ for some finite constants τ, K_1, and K_2, independent of u.

Theorem 2 If $\delta > 0$, there exists $\varepsilon_0 > 0$ such that, whenever $u \in \mathcal{U}$ and $0 \le \varepsilon < \varepsilon_0$, the respective outputs y and y_ε of (1) and (2) satisfy

$$\max_{0 \le t \le \tau} \|y(t) - y_\varepsilon(t)\| < \delta.$$

Interpreting the number δ as the minimum achievable measurement error, Theorem 2 shows that the nominal model (A,B,C,D) and the perturbed model (1) are equivalent in an input-output sense for sufficiently small ε. An analogous statement applies to the compensator. Hence, Theorems 1 and 2 combine to show that, if only input-output information is available in the plant and compensator models, stable closed-loop behavior can never be guaranteed.

[1] M. Vidyasagar, "Robust Stabilization of Singularly Perturbed Systems," *Systems and Control Letters*, 5 (1985), 413-418.

[2] H. K. Khalil, "A Further Note on the Robustness of Feedback Control Methods to Modelling Errors," *IEEE Trans. Auto. Control*, 29 (1984), 403-418.

[3] C. E. Rohrs, L. Valvani, M. Athans, G. Stein, "Robustness of Continuous-Time Adaptive Control Algorithms in the Presence of Unmodelled Dynamics," *IEEE Trans. Auto. Control*, (30) 1985, 881-889.

[4] P. Ioannou, P. V. Kokotovic, "Singular Perturbations and Robust Redesign of Adaptive Control," *Proc. 21st IEEE Conf. on Decision and Control*," Dec. 1982, 24-29.

A FUNCTIONAL ANALYSIS APPROACH TO AUTOMATIC GAIN CONTROL

DOUGLAS COCHRAN
Harvard University
Cambridge, Massachusetts

Abstract

An ideal automatic gain control (AGC) may be described as a signal processing operation that is insensitive to re-scaling of its input. Functions whose values do not change when their arguments are multiplied by a scalar are known in functional analysis as homogeneous of degree zero (HDZ) maps. The notion of HDZ maps is used as a starting point for studying AGC and explaining some of the difficulties encountered in attempting to apply traditional techniques in analysis of signal processing operations involving AGC.

MATHEMATICAL FORMULATION OF IDEAL AGC

In this note, signals will be regarded as elements in a (real or complex) Banach space. With this setting, a natural starting point for studying automatic gain control is the following definition from functional analysis:

> **Definition**: A map T between Banach spaces X and Y is homogeneous of degree k if $T(a\mathbf{x}) = a^kT(\mathbf{x})$ for all $\mathbf{x}$ in X and all non-zero scalars a.

In particular, T is homogeneous of degree zero (HDZ) if $T(a\mathbf{x}) = T(\mathbf{x})$ for all $\mathbf{x}$ and non-zero a. This definition is pertinent to the gain control situation since, in this setting, mapping of a signal in X by a homogeneous of degree zero function is a signal processing operation that is insensitive to adjustment of the input signal's gain. It is, however, slightly too restrictive to characterize ideal automatic gain control because many commonly-used normalization methods allow the phase (or sign) of the output signal to change while holding its magnitude constant if the gain on the input signal is adjusted. To capture this phenomenon, the following more general notion of an HDZ map will be used:

> **Definition**: A map T between Banach spaces X and Y is homogeneous of generalized degree zero (HGDZ) if $T(a\mathbf{x}) = \beta T(\mathbf{x})$ where $|\beta|=1$ for all $\mathbf{x}$ in X and all non-zero scalars a.

Any HDZ map is trivially HGDZ, but the concepts are clearly not equivalent. Given the purpose of AGC, the following definition seems appropriate:

> **Definition:** Suppose X and Y are Banach spaces whose elements are called signals. Then a signal processing operation which maps signals

in X to signals in Y exhibits ideal automatic gain control if it is HGDZ as a map of Banach spaces.

CHARACTERISTICS OF IDEAL AGC

If $T:\mathbf{R}\to\mathbf{R}$ is a HDZ map, then $T(kx) = T(x)$ for all real x and non-zero k. In particular, $T(x\cdot x^{-1}) = T(1) = T(x)$ for all x so that T must be a constant function. It is tempting to draw the impression from this example that the class of HDZ functions is narrow. This is not true, even in the case of real Banach spaces. As an example, consider $T:\mathbf{R}^2\to\mathbf{R}$. Since $T(kx,ky) = T(x,y)$, T takes the same value at any point on a line through the origin. The class of HDZ maps on $\mathbf{R}^2$ is thus identified with the class of all maps on the projective space $\mathbf{RP}^1$; similarly, any function on $\mathbf{RP}^{n-1}$ has a corresponding HDZ map on $\mathbf{R}^n$. There are at least as many HGDZ maps as HDZ maps on any space and, if the scalars are complex, the class of HGDZ functions is further diversified by the relative weakness of the condition $T(a\mathbf{x}) = e^{i\theta}T(\mathbf{x})$.

While the above demonstrates there is no shortage of potential signal processing algorithms exhibiting ideal AGC, the following facts show that they all have some properties that make them somewhat difficult to analyze.

Proposition: The only linear signal processing algorithm exhibiting ideal AGC is the trivial map, $T(\mathbf{x})=0$ for all $\mathbf{x}$.

Proof: If T is linear, $T(2\mathbf{x})=2T(\mathbf{x})$. If T is HGDZ, $T(2\mathbf{x}) = \alpha T(\mathbf{x})$ with $|\alpha|=1$. But $|2|$ is not 1, so $T(\mathbf{x})=0$ for all $\mathbf{x}$.

Proposition: If the (Frechet) derivative of a non-constant HDZ map T on a Banach space X exists at zero, it is unbounded (discontinuous).

Proof: If the derivative DT_0 of T at 0 exists, its values are $DT_0(\mathbf{x}) = \lim \epsilon^{-1}[T(0+\epsilon\mathbf{x}) - T(0)] = \lim \epsilon^{-1}[T(\epsilon\mathbf{x})-T(0)] = \lim \epsilon^{-1}[T(\mathbf{x})-T(0)]$ where the limit is as $\epsilon\to 0$. Since T is non-constant, there is some $\mathbf{x}_0$ in X with $T(\mathbf{x}_0)$ not equal to $T(0)$.

These properties suggest that AGC algorithms may be difficult to analyze by traditional techniques -- not only are they non-linear, they do not generally admit well-behaved linear approximations. Despite the ubiquity of signal processing algorithms exhibiting AGC in common use, the fact that all such techniques are necessarily non-linear is often ignored when it comes to designing their characteristics and explaining their behavior.

APPLICATION OF THE ROBUST SERVOMECHANISM CONTROLLER TO SYSTEMS WITH PERIODIC TRACKING/DISTURBANCE SIGNALS

E.J. DAVISON, P. PATEL
Department of Electrical Engineering
University of Toronto
Toronto, Ontario, Canada M5S 1A4

ABSTRACT

The following problem is considered. Given a system which has tracking/disturbance signals arising from a class of periodic signals with period T say, assume that such a periodic signal has p dominant harmonic components contained in $\{\omega_l, l=0,1,...,\nu\}$. Then it is desired to find a finite dimensional linear controller which gives exact robust asymptotic regulation for the case of sinusoidal signals with frequencies ω_l, $l=0,1,...,\nu$, subject to certain controller gain magnitude constraints or/and gain margin tolerance constraints which may be imposed on the problem. This is called "the approximate robust servomechanism problem for periodic signals of the class $\{\omega_l, l=0,1,...,\nu\}$". It is to be noted that such a problem cannot in general be solved by using high gain control synthesis methods. Existence conditions and a controller design method are given to solve the problem. A number of examples are given to illustrate the type of results that may be achieved.

1. INTRODUCTION

The following type of problem is considered in this paper. Given a finite dimensional linear time-invariant multivariable system, suppose that the tracking/disturbance signals arise from a class of periodic signals with period T say, e.g. a triangular periodic wave. Then, unless the periodic signal $\omega(t)$ arises from the following finite dimensional linear model: $\dot{\eta}=\mathbf{A}\eta$, $\omega=\mathbf{C}\eta$, in general, there exists no finite dimensional linear time-invariant controller which can be applied to the system to produce exact robust asymptotic error regulation. However, it may be desired to obtain a finite dimensional controller so that approximate robust asymptotic error regulation occurs. In this case, if one assumes that the periodic signal possesses a Fourier series which has p dominant harmonic components contained in $\{\omega_l, l=0,1,...,\nu\}$, "approximate p mode error regulation" for the system is said to occur if there exists a finite dimensional linear controller which produces exact robust asymptotic error regulation for linear controller which produces exact robust asymptotic error regulation for sinusoidal tracking/disturbance signals of frequency ω_l, $l=0,1,2,...,\nu$.

In solving this type of approximate problem, the following observations are to be noted:

(a) Any realistic controlled system always has restrictions imposed on the magnitude of controller gain elements which can be applied (e.g. due to possible nonlinear effects due to actuators, due to possible closed loop instability which may arise

---

This work has been supported by the Natural Sciences and Engineering Research Council of Canada under grant no. A4396.

from modelling errors, etc.).

(b) Generally, it will be the case, that at least some of the frequencies ω_l, $l=0,1,2,\ldots,\nu$ which are to be regulated, are contained in the bandwidth of the system to be controlled.

(c) The system may or may not be minimum phase.

Thus, these observations give rise to the following problem: given $\{\omega_l,\ l=0,1,\ldots,\nu\}$, find a controller, in particular, the "simplest possible" controller, i.e. one which has minimum order, which will give exact robust error regulation for the system with disturbance/tracking signals of frequencies ω_l, $l=0,1,\ldots,\nu$, subject to specified engineering constraints imposed on the magnitude of controller gains [1] and gain margin tolerance [2] requirements, say. This is called the "approximate RSP for periodic systems of the class $\{\omega_l,\ l=0,1,\ldots,\nu\}$".

This problem can be directly solved by treating the problem as a robust servomechanism problem using the parameter optimization method of [1]. It is to be emphasized that in general it is impossible to satisfy the above requirements by demanding that "good" regulation occurs $\forall\omega\in[0,\omega_{\max}]$, $\omega_{\max}\triangleq\max\{\omega_l,\ l=0,1,\ldots,\nu\}$, since arbitrarily large controller gains are then required to satisfy this requirement.

It is the purpose of this paper to present some case studies of such a design method when applied to this type of problem. An alternative design method to this type of problem has been studied in [8] for single-input/single-output systems using an infinite dimensional compensator; in this case no gain constraints are imposed and only sufficient conditions for a solution to exist are obtained.

2. DEVELOPMENT

Consider the linear time-invariant system:

$$\begin{aligned} \dot{x} &= Ax + Bu + E\omega \\ y &= Cx + Du + F\omega \quad ; \quad e \triangleq y - y_{ref} \\ y_m &= C_m x + D_m u + F_m \omega \end{aligned} \tag{1}$$

where $x\in R^n$ is the state, $u\in R^m$ the inputs, $y\in R^r$ the outputs to be regulated, $y_m\in R^{r_m}$ are the measurable outputs, $\omega\in R^{\Omega}$ is a disturbance, and $y_{ref}\in R^r$ is the reference-input signal.

Assume that the elements of ω are y_{ref} are bounded periodic signals with period T, which satisfy the Dirichlet conditions, so that they possess a Fourier series. In particular, assume that the first p dominant harmonic components of the Fourier series for each element of ω and y_{ref} are contained in the following $\nu\geq p$ harmonic components:

$$\omega_l \triangleq \frac{2l\pi}{T}\ , \quad l=0,1,\ldots,\nu \tag{2}$$

Then, if there exists a solution to the robust servomechanism problem (RSP) [1] for (1) for the case of disturbance/tracking signals poles $\pm j\omega_l$, $l=0,1,2,\ldots,\nu$ [1], it is said that there exists an approximate p mode solution to the robust servomechanism problem for (1), for the class of periodic signals with dominant harmonic content $(\omega_0,\omega_1,\ldots,\omega_\nu)$, or more briefly, that *there exists an approximate solution to the RSP for periodic signals of the class* $\{\omega_l,\ l=0,1,2,\ldots,\nu\}$.

The following result on the existence of a solution to the above problem is immediately obtained from [1]:

Lemma 1: Given (1), there exists an approximate solution to the RSP for periodic signals of the class $\{\omega_l, l=0,1,2,\ldots,\nu\}$ iff the following conditions are all satisfied:

(a) (C_m, A, B) is stabilizable and detectable.

(b) (C, A, B, D) has no transmission zeros located at $\pm j\omega_l$, $l=0,1,2,\ldots,\nu$,

(c) $m \geq r$,

(d) $y \subset y_m$, i.e. the outputs to be regulated are measurable.

Assume that lemma 1 holds; then a controller which solves the RSP consists of two parts (i) a servo-compensator and (ii) a stabilizing compensator. In the case when the system (1) is open loop asymptotically stable, then there always exists a stabilizing compensator of order zero [1]; in this case, the minimal order controller which solves the RSP for periodic signals of the class $\{\omega_l, l=0,1,\ldots,\nu\}$ is given as follows [1]:

$$u = \hat{K}_0 \hat{y}_m + K\eta \ , \quad \hat{y}_m \overset{\Delta}{=} y_m - D_m u \tag{3}$$

where $\eta \in R^{r(\nu+1)}$ is the output of the servo-compensator given by:

$$\dot{\eta} = \text{block diag}\left\{ \begin{pmatrix} 0 & I_r \\ -\omega_0^2 I_r & 0 \end{pmatrix}, \begin{pmatrix} 0 & I_r \\ -\omega_1^2 I_r & 0 \end{pmatrix}, \cdots, \begin{pmatrix} 0 & I_r \\ -\omega_\nu^2 I_r & 0 \end{pmatrix} \right\} \eta + \left[\begin{array}{c} O_r \\ I_r \\ \hline O_r \\ I_r \\ \hline \vdots \\ \hline O_r \\ I_r \end{array}\right] (y - y_{ref}) \tag{4}$$

where $K \overset{\Delta}{=} (K_0, K_1, \ldots, K_\nu)$, and O_r denotes a $r \times r$ zero matrix.

The following result is now directly obtained from [1] re the existence of a controller which solves the RSP for periodic signals, subject to controller gain constraints [1] and gain margin/time lag tolerance constraints [2] imposed on the closed loop system.

Theorem 1: Consider the system (1) and assume that A is asymptotically stable; then there exists a controller which solves the approximate RSP for periodic signals of the class $\{\omega_l, l=0,1,\ldots,\nu\}$ such that:

(i) the controller gain elements satisfy any specified magnitude constraint $k_{\max} > 0$, i.e. $\|\hat{K}, K\| < k_{\max}$, where $\|\cdot\|$ is any specified norm,

(ii) the resultant closed loop system satisfies any specified gain margin or/and time lag tolerance constraints [2],

if and only if conditions (b), (c), (d) of lemma 1 all hold. Moreover, if these conditions are all satisfied, then a controller which solves the problem is given by (3).

Thus if a stable system satisfies lemma 1, this implies that a controller with the structure of (3) can always be used to solve the approximate RSP for periodic systems. A method of designing such a controller which gives maximum speed of response, low interaction [3], and which satisfies the two constraints (i), (ii) above, may be obtained by minimizing a performance index of the "cheap control" type with respect to the controller parameters $\hat{K}$, K by using nonlinear programming methods, subject to the gain margin, etc. and controller magnitude constraints as outlined in [1]. In this case, the performance index chosen for the problem is given by:

$$J = E \int_0^\infty (z' Qz + \epsilon u' Ru) d\tau \tag{5}$$

where $Q > 0$, $R > 0$, $\epsilon \to 0$ and $z \in R^r$ is the output of the servo-compensator (4), i.e.

$$z \triangleq [I_r, O_r, I_r, O_r, \dots, I_r, O_r]\eta \tag{6}$$

In order to carry out this parameter optimization problem, it is necessary to have a feasible initial starting point. Such an initial feasible starting point can be obtained as follows.

In (3), let:

$$K \triangleq \{K_{01}, K_{02}; K_{11}, K_{12}; \dots; K_{\nu 1}, K_{\nu 2}\} \tag{7}$$

where $K_{ij} \in R^{m \times r}$, $\forall i,j$, and given $\{\omega_l, l=0,1,\dots,\nu\}$ where $\omega_0 = 0$, $\omega_i > 0$, $i=1,\dots,\nu$, define $\mathbf{H}_i \in R^{2m \times 2r}$ as follows:

$$\mathbf{H}_i = \left[\begin{pmatrix} \mathbf{C}_i & 0 \\ 0 & \mathbf{C}_i \end{pmatrix} \begin{pmatrix} \mathbf{A}_i & \omega_i I \\ -\omega_i I & \mathbf{A}_i \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{B}_i & 0 \\ 0 & \mathbf{B}_i \end{pmatrix}\right]^+ , \quad i = 1,2,\dots,\nu \tag{8}$$

where if A is stable and lemma 1 holds all quantities are well defined [4], where $\mathbf{A}_i \in R^{(n+2r_i)\times(n+2r_i)}$, $i=1,2,\dots,\nu+1$ is defined recursively as follows:

$$\mathbf{A}_i \triangleq \begin{bmatrix} \mathbf{A}_{i-1} & \mathbf{B}_{i-1}K_{i-1,1} & \mathbf{B}_{i-1}K_{i-1,2} \\ 0 & 0 & I_r \\ \mathbf{C}_{i-1} & -\omega_{i-1}^2 I_r & 0 \end{bmatrix} \tag{9}$$

where $\mathbf{B}_i \in R^{(n+2r_i)\times m}$ and $\mathbf{C}_i \in R^{r \times (n+2r_i)}$, $i=1,2,\dots,\nu$ are defined by

$$\mathbf{B}_i \triangleq \begin{pmatrix} B \\ 0 \end{pmatrix}, \quad \mathbf{C}_i \triangleq (C, 0) \tag{10}$$

and

$$\mathbf{A}_0 \triangleq A \ , \quad \mathbf{B}_0 \triangleq B \ , \quad \mathbf{C}_0 \triangleq C$$

The following lemma is easily established by direct evaluation.

Lemma 2: Given $\mathbf{C}_i$, $\mathbf{A}_i$, $\mathbf{B}_i$ defined in (9), (10), assume that lemma 1 holds; then rank $\mathbf{H}_i = 2r$, $i=1,2,\dots,\nu$ and on defining

$$\begin{pmatrix} \Phi_1^i & \Phi_2^i \\ -\Phi_2^i & \Phi_1^i \end{pmatrix} \triangleq \mathbf{H}_i$$

then

$$\Phi_1^i = \mathrm{Re}\{\mathbf{C}_i(\mathbf{A}_i + j\omega_i I)^{-1}\mathbf{B}_i\}^+$$
$$\Phi_2^i = \mathrm{Im}\{\mathbf{C}_i(\mathbf{A}_i + j\omega_i I)^{-1}\mathbf{B}_i\}^+$$

for $i=1,2,\dots,\nu$.

The following result is now directly obtained from [4] and lemma 2.

Theorem 2: Given the asymptotically stable system (1), assume that lemma 1 holds, that a gain margin or/and time lag tolerance constraint, and controller magnitude gain constraint is specified. Then there always exist scalars $\bar{\epsilon}_{01}>0$, $\bar{\epsilon}_{02}>0$, $\bar{\epsilon}_i>0$, $i=1,2,\ldots,\nu$ such that the controller (3) solves the approximate RSP for periodic signals of the class ω_l, $l=0,1,\ldots,\nu$, using the following controller gains:

$$\begin{aligned}
&\hat{K} \triangleq 0\\
&K_{02} \triangleq \bar{\epsilon}_{01}(CA^{-1}B)^{+}\\
&K_{01} \triangleq \bar{\epsilon}_{02}\bar{\epsilon}_{01}(CA^{-1}B)^{+}\\
&\left.\begin{aligned}
K_{i2} &\triangleq 2\bar{\epsilon}_i(I_m \quad 0)\mathbf{H}_i\\
K_{i1} &\triangleq 2\bar{\epsilon}_i\omega_i(0 \quad I_m)\mathbf{H}_i
\end{aligned}\right\} i=1,2,\ldots,\nu
\end{aligned} \tag{11}$$

where $\mathbf{H}_i$ is defined in (8).

This lemma forms the basis of the following algorithm which can be used to find an initial feasible starting point for the parameter optimization problem (5). The following definition is made. Given $(\cdot)\in R^{n\times n}$, let $\mathrm{Re}[\lambda(\cdot)] \triangleq \max\limits_{j=1,2,\ldots,n}\{\mathrm{Re}(\lambda_j(\cdot))\}$, where λ_j, $j=1,\ldots,n$ are the eigenvalues of $(\cdot)$.

Algorithm 1:

1. Put $\hat{K}=0$ and find $\epsilon_{o1}>0$ using a 1-dimensional search to stabilize $\begin{Bmatrix} A & BK_{02}\\ C & 0\end{Bmatrix}$ such that $\min\limits_{\epsilon_{01}} \mathrm{Re}\left[\lambda\begin{Bmatrix} A & BK_{02}\\ C & 0\end{Bmatrix}\right]$ occurs, subject to the gain margin and controller gain magnitude constraint. Fix ϵ_{01}.
2. Find $\epsilon_{02}>0$ using a 1-dimensional search to stabilize $\mathbf{A}_1$ such that $\min\limits_{\epsilon_{02}}\mathrm{Re}[\lambda(\mathbf{A}_1)]$ occurs, subject to the two constraints. Fix ϵ_{02}.
3. Find $\epsilon_1>0$ using a 1-dimensional search to stabilize $\mathbf{A}_2$ such that $\min\limits_{\epsilon_1}\mathrm{Re}[\lambda(\mathbf{A}_2)]$ occurs, subject to the two constraints. Fix ϵ_1.

⋮

ν. Find $\epsilon_\nu>0$ using a 1-dimensional search to stabilize $\mathbf{A}_{\nu+1}$ such that $\min\limits_{\epsilon_\nu}\mathrm{Re}[\lambda(\mathbf{A}_{\nu+1})]$ occurs subject to the two constraints. Fix ϵ_ν.

It follows from [4], that all steps of this algorithm can always be carried out.

3. NUMERICAL EXAMPLES

In the case studies to follow, the choice of $Q=I$, $R=I$, $\epsilon=10^{-8}$ in the performance index (5) was used in all examples. The algorithm of [1], [5] was used to carry out the parameter optimization procedure for (5). In all simulations reported, zero initial conditions are assumed on the plant and controller.

Example 1 (DC Motor): Consider the model of a DC motor described as follows [1]

$$\dot{x} = \begin{bmatrix} -\frac{B}{J} & -\frac{K_T}{J} \\ -\frac{K_e}{L} & -\frac{R}{L} \end{bmatrix} x + \begin{pmatrix} 0 \\ \frac{1}{L} \end{pmatrix} u + \begin{pmatrix} -\frac{1}{J} \\ 0 \end{pmatrix} \omega \tag{12}$$

$$y = (1 \quad 0)x$$

where B=0.0162, J=0.215, K_T=1.11, K_e=1.11, R=1.05 and L=0.0053. This system has open loop eigenvalues (-5.70, -192.5) and no transmission zeros so that lemma 1 holds.

Assume it is desired to solve the approximate RSP for periodic signals of the class $(0,\pi,3\pi,5\pi,7\pi,9\pi)$, using the controller (3), such that all elements of the gain matrices $(\hat{K}_0,K)$ satisfy the magnitude constraint $|k_{ij}| \leq 10$, $\forall i,j$.

The following solution is obtained on solving the parameter optimization problem (5) subject to this constraint:

$J_{\text{optimal}} = 5.36$

Closed loop eigenvalues $|_{\text{optimal}} = \{-6.26,-181,-2.38\pm j1.61,-1.01\pm j2.76,-0.83\pm j9.6, -0.54\pm j16.1,-0.42\pm j22.5,-0.30\pm j28.7\}$

Controller gains $|_{\text{optimal}}$

$\hat{K}_0 = -2.26$

$K = (K_0,K_1,K_2,K_3,K_4,K_5)$

where

$K_0 = (-9.999,-9.999)$

$K_1 = (-9.999,-3.8)$

$K_2 = (3.8,4.7)$

$K_3 = (9.4,-4.8)$

$K_4 = (9.999,-5.7)$

$K_5 = (9.7,-5.5)$

In this case, there are 4 active constraints occurring in the constrained optimization problem.

Some typical simulations of the resultant closed loop system are given in Figure 1 for tracking a triangular periodic signal, and in Figure 2 for regulating the system against two types of triangular periodic disturbances. It is seen that excellent tracking/disturbance rejection occurs.

A frequency response plot of the closed loop and open loop system's transfer function between the output y and input ω is given in Figures 3, 4 respectively. Also, a frequency response plot of the closed loop and open loop system's transfer function between the output y and input y_{ref}, u respectively is given in Figures 5, 6. It is seen from these plots, that the effect of the controller (3) is to selectively cause exact error attenuation to occur at the selected frequencies $(0,\pi,3\pi,5\pi,7\pi,9\pi)$.

Example 2 (Head Box): Consider the model of a head box described as follows [6]:

$$\dot{x} = \begin{pmatrix} -0.395 & 0.01145 \\ -0.011 & 0 \end{pmatrix} x + \begin{pmatrix} 0.3362 & 1.038 \\ 0.000966 & 0 \end{pmatrix} u + \begin{pmatrix} 1 \\ 1 \end{pmatrix} \omega \qquad (13)$$

$$y = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} x$$

This system has open loop eigenvalues $(-3.2_{10^{-4}}, -3.9_{10^{-1}})$ and no transmission zeros so that lemma 1 holds.

Assume it is desired to solve the approximate RSP for periodic signals of the class $(\pi, 3\pi, 5\pi)$ using the controller (3), such that the resultant closed loop system has a gain margin tolerance of (0.5,1.5).

The following solution is obtained on solving the parameter optimization problem (5) subject to this constraint.

$$J_{\text{optimal}} = 70.8$$

$$\left.\begin{matrix}\text{closed loop} \\ \text{eigenvalues}\end{matrix}\right|_{\text{optimal}} = \begin{matrix}\{-40.1, -254, -0.46 \pm j3.22, -1.20 \pm j3.04, -0.093 \pm j9.43, \\ -1.41 \pm j9.2, -0.045 \pm j15.7, -1.23 \pm j15.1\}\end{matrix}$$

Controller gains $|_{\text{optimal}}$

$$\hat{K}_0 = \begin{pmatrix} -2.39_{10^3} & -4.32_{10^3} \\ -2.09_{10^2} & 3.87_{10^3} \end{pmatrix}$$

$$K = (K_1, K_2, K_3),$$

where

$$K_1 = \begin{pmatrix} -5.84_{10^2} & 2.05_{10^3} & -5.02_{10^3} & -4.18_{10^3} \\ -1.20_{10^2} & 1.28_{10^3} & -8.63_{10^2} & 3.52_{10^3} \end{pmatrix}$$

$$K_2 = \begin{pmatrix} -1.25_{10^3} & -2.59_{10^3} & -6.28_{10^3} & -7.90_{10^2} \\ -1.21_{10^3} & -7.97_{10^3} & -4.56_{10^2} & 7.65_{10^2} \end{pmatrix}$$

$$K_3 = \begin{pmatrix} -1.32_{10^3} & -1.48_{10^3} & -7.05_{10^3} & -3.93_{10^2} \\ 2.60_{10^2} & -2.23_{10^3} & -5.63_{10^2} & 3.39_{10^2} \end{pmatrix}$$

In this case, there is one active constraint occurring in the constrained optimization problem.

Some typical simulations of the resultant closed loop system are given in Figure 7 for tracking a rectangular periodic signal, in Figure 8 for tracking a triangular periodic signal, and in Figure 9 for regulating the system against a triangular periodic disturbance signal. It is seen that excellent tracking/disturbance rejection occur. In particular, almost no interaction occurs in the tracking cases.

Example 3: (Unstable Batch Reactor)

Consider the model of a batch reactor described as follows [7]:

$$\dot{x} = \begin{bmatrix} 1.38 & -0.2077 & 6.715 & -5.676 \\ -0.5814 & -4.29 & 0 & 0.675 \\ 1.067 & 4.273 & -6.654 & 5.893 \\ 0.048 & 4.273 & 1.343 & -2.104 \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ 5.679 & 0 \\ 1.136 & -3.146 \\ 1.136 & 0 \end{bmatrix} u + \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \omega$$

$$y=\begin{bmatrix}1 & 0 & 1 & -1\\ 0 & 1 & 0 & 0\end{bmatrix}x \tag{14}$$

This system has open loop eigenvalues (+1.99,+0.064,-5.06,-8.67) and transmission zeros (-1.192,-5.039) so that lemma 1 holds, but the system is not open loop stable.

Assume that it is desired to solve the approximate RSP for periodic signals of the class $(,\pi,3\pi,5\pi)$ using the controller (3) with no constraints imposed; in this case there is no guarantee that such a controller will solve the problem. However, on noting that the system (14) can be stabilized by using the control $u=\hat{K}_0 y$ where $\hat{K}_0=\begin{pmatrix}0 & -2\\ 5 & 0\end{pmatrix}$, then in fact such a controller will solve the problem.

The following solution is obtained on solving the parameter optimization problem (5) on using the controller $u=\begin{pmatrix}0 & -2\\ 5 & 0\end{pmatrix}y$ to prestabilize the system (14):

$$J_{\text{optimal}}=3.41$$

$$\text{closed loop eigenvalues}\,\Big|_{\text{optimal}}=\begin{array}{l}\{-1.22,-5.28,-4.26\pm j4.75,-0.901\pm j2.55,-1.19\pm j2.44,\\ -1.76\pm j7.21,-1.75\pm j9.49,-2.00\pm j11.0,-1.53\pm j15.9,\\ -2.02\pm j15.1,-2.00\pm j27.0\}\end{array}$$

Controller gains $|_{\text{optimal}}$

$$\hat{K}_0=\begin{pmatrix}-4.47 & -0.145\\ 9.16 & -7.61\end{pmatrix}$$

$$K=(K_0,K_1,K_2,K_3)$$

where

$$K_0=\begin{pmatrix}-276 & -68 & -111 & -42\\ 421 & -188 & 190 & -79\end{pmatrix}$$

$$K_1=\begin{pmatrix}-163 & -54 & 44 & 10\\ 241 & -182 & -65 & 21\end{pmatrix}$$

$$K_2=\begin{pmatrix}-161 & -60 & -1.2 & -0.16\\ 176 & -170 & 7.8 & -6.8\end{pmatrix}$$

$$K_3=\begin{pmatrix}-228 & -61 & -5.3 & 3.0\\ 188 & -52 & 15 & -15\end{pmatrix}$$

Some typical simulations of the resultant closed loop system are given in Figure 10 for tracking a triangular periodic wave, and in Figures 11, 12 for regulating the system against a coloured noise input with cut off frequency $\theta=5\pi$, 15π rad/sec respectively. To generate this noise, a 10th order Butterworth filter with cut off frequency θ, and with white noise input of mean 0.5 and distribution between 0 and 1, was used. It is seen that excellent regulation occurs in all cases.

4. CONCLUSIONS

This paper has considered the problem of achieving approximate robust tracking/regulation for a class of periodic signals which have a given period, by using a finite dimensional linear controller. This has been done by defining and solving "the approximate RSP for periodic signals of the class $\{\omega_l, l=0,1,...,\nu\}$". In this problem formulation, it is assumed that the controller gains are constrained either directly by a controller gain magnitude constraint, or indirectly, by means of a gain margin constraint imposed on the resultant closed loop system. Thus, this implies that any design metehod which uses a high gain type of approach, e.g. the H_∞ approach, cannot be directly applied. Three case studies of the design method are given to illustrate the type of results that may be obtained; in all cases, excellent tracking/regulation occurs.

5. REFERENCES

[1] Davison, E.J., Ferguson, I., "Design of controllers for the multivariable robust servomechanism problem using parameter optimization methods", *IEEE Trans. on Auto. Control*, vol. AC-26, no. 1, 1981, pp. 93-110.

[2] Davison, E.J., Copeland, B., "Solution of the robust servomechanism problem subject to specified gain margin and time lag tolerance constraints", *IEEE Trans. on Auto Control*, vol. AC-30, no. 3, pp. 229-239.

[3] Scherzinger, B., Davison, E.J., "Approximate error regulation - the robust servomechanism problem for unknown input signals", *21st IEEE CDC*, Dec. 1982, pp. 871-876.

[4] Davison, E.J., "Multivariable tuning regulators: the feedforward and robust control of a general servomechanism problem", *IEEE Trans. on Auto. Control*, vol. AC-21, no. 1, 1976, pp. 35-47.

[5] Gesing, W., Davison, E.J., "An exact penalty function algorithm for solving general constrained parameter optimization problems", *Automatica*, vol. 15, 1979, pp. 175-188.

[6] MacFarlane, A.G., Belletruttie, J., "The characteristic locus design method", *Automatica*, vol. 9, 1973, pp. 575-588.

[7] Rosenbrock, H.H., *Computer-Aided Control System Design*, Academic Press, 1974.

[8] Hara, S., Omata, T., Nakano, M., "Synthesis of repetitive control systems and its application", *1985 IEEE CDC*, Dec. 1985, pp. 1387-1392.

*** The figures are omitted due to page limitation. They may be obtained by writing to the authors.

MULTI-TIME SCALE CANONICAL FORM AND HIGH-GAIN REGULATOR PROBLEM

Y. ANDO* M. SUZUKI** AND K. HAMADA**
* Nagoya Municipal Industrial Research Institute, NAGOYA, JAPAN
** Nagoya University, NAGOYA, JAPAN

1.INTRODUCTION

It has been known that a high-gain feedback is very effective to reduce the parameter sensitivities, the effect of disturbances and so on. One of the best methods to analyze and to design the high-gain feedback systems is a singular perturbation method. Young et al. [3] introduced the singular perturbation method at first. Sannuti et al. [4] expanded applications into multi-time scale systems. We consider the high-gain regulator problem (HRP) as below.

2.High-gain regulator problem(HRP)

We consider a system

$$\dot{x} = Ax + Bu, \tag{1}$$
$$y = Cx, \tag{2}$$

where $x \in R^n$, $u \in R^r$ and $y \in R^m$. We introduce the performance index

$$J = \int_0^\infty y^T y \, dt. \tag{3}$$

We say that HRP is solvable if there exist a high-gain feedback law

$$u = K(g_0, g_1, \ldots, g_k)x \tag{4}$$

so that $A+BK(g_0, g_1, \ldots, g_k)$ is stable and

$$\lim_{\substack{g_i \to \infty \\ i=0,\cdots,k}} J = 0 \ , \tag{5}$$

where $g_1, \ldots, g_k$ $(g_1 > g_2 > \ldots > g_k)$ are sufficiently large gain scalar parameters which define multi-time scales.

First we consider a simple control law with one gain parameter

$$u = gKx. \tag{6}$$

Then following theorem is established.

THEOREM 1 When one parameter high-gain feedback(6) is adopted, HRP is solvable if conditions (i)-(iii) are satisfied.

(i) $r > m$
(ii) rank $CB = m$
(iii) rank $\begin{bmatrix} sI-A & B \\ C & 0 \end{bmatrix} = n+m$ for all s with $\mathrm{Re}(s) > 0$

3.MULTI-TIME SCALE CANONICAL FORM

In the next, we consider the multi-time scale high-gain systems. Before analysis we introduce the multi-time scale canonical form. At first we set that $A_0=A$, $B_0=B$ and $x_0=x$. We transform A_i and B_i such that

$$M_i B_i T_i = \begin{bmatrix} 0 & 0 \\ F_{i+1} & 0 \end{bmatrix} \tag{7}$$

$$M_i A_i S_i = \begin{bmatrix} A_{i+1} & B_{i+1} \\ C_{i+1} & D_{i+1} \end{bmatrix} \tag{8}$$

where M_i, T_i and F_{i+1} are nonsingular matrices and S_i is an inverse matrix of M_i. This transformations can be iterated and end within n-times when matrix B_i is a nonsingular or zero matrix. Simple consideration leads to following lemma.

LEMMA 1 (i) If and only if transformations end with nonsingular B_i, then original system (1) is controllable.

(ii) If transformations end with $B_i=0$, then if and only if A_i is stable, then original system (1) is stabilizable.

With these transformations we get the multi-time scale canonical form representation

$$\begin{bmatrix} \dot{x}_k \\ \dot{v}_k \\ \vdots \\ \dot{v}_1 \\ \dot{v}_0 \end{bmatrix} = \begin{bmatrix} A_{k+1} & B_{k+1} & 0 & 0 \cdots & 0 \\ C_{k+1} & D_{k+1} & F_{k+1} & 0 \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ G_{1,k+1} & G_{1,k} & G_{1,k-1} & F_2 & 0 \\ G_{0,k+1} & G_{0,k} & G_{0,k-1} & G_{0,1} & \end{bmatrix} \begin{bmatrix} x_k \\ v_k \\ \vdots \\ v_1 \\ v_0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ F_1 \end{bmatrix} u \tag{9}$$

and

$$x = \widehat{R}_k x_k + R_k v_k + \cdots + R_2 v_2 + R_1 v_1 + R_0 v_0, \tag{10}$$

where v_i is an r_i-dimensional vector with r_i=rank B_i and x_k is an $(n-\sum_{i=0}^{k} r_i)$-dimensional vector. Coefficient matrices R_i's span a subspace such that

$$\mathrm{Range}[R_0\ R_1\ \ldots\ R_i] = \mathrm{Range}[B\ AB\ A^2B\ \ldots\ A^iB]. \tag{11}$$

Thus this representation shows how the input u influences the state space through range(B) ,range(AB), range(A^2B) and so on.

As a multi-time scale high-gain feedback law we adopt

$$u=g_0K_0(v_0+g_1K_1(\ldots g_{k-1}K_{k-1}(v_{k-1}+g_kK_k(v_k+\widehat{K}_kx_k))\ldots)), \tag{12}$$

where K_i is chosen in order to make K_iB_i stable. If each g_i satisfies to module of stability of singular perturbation method, then

$$x = \sum_{i=0}^{k}(R_iv_i) + \widehat{R}_kx_k \tag{13}$$

where v_k tends to zero with speed of order g_k. Therefor when each g_i tends to infinity, x is approximated by R_kx_k.

THEOREM 2 It is assumed that original system is stabilizable. HRP is solvable if there exists an integer k such that

$$\mathrm{rank}\ [CB\ CAB\ \ldots\ CA^kB] = m. \tag{14}$$

4.CONCLUSION

The results which are got above show that conditions for solvability of high-gain regulator problem are able to check easily only by easy matrix operations. In THEOREM 1 a number of time scale is arranged before design and in THEOREM 2 necessary numbers of time scale is defined from design. We will show how many time scales are sufficient to solve HRP and what conditions are necessary and sufficient to solve HRP.

<REFERENCE>

[1] H. Kimura : A New Approach to Perfect Regulation and Bounded Peaking in Linear Multivariavble Control Systems, IEEE Trans. on Automatic Control, AC-26-1, 253/270

[2] J.C. Willems : Almost Invariant Subspaces : An approach to High Gain Feedback Design Part 1 : Almost Controlled Invariant Subspaces, IEEE Trans. on Automatic Control, AC-26-1, 235/252

[3] K.D. Young , P.V. Kokotovic and V.I. Utkin : Singular Perturbation Analysis of High-Gain Feedback Systems, IEEE Trans. on Automatic Control, AC-22-6, 931/937

[4] P. Sannuti and H.S. Wason : Multiple Time-Scale Decomposition in Cheap Control Problem - Singular Control,IEEE Trans. on Automatic Control, AC-30-7,633/644

ANALYSIS AND DESIGN OF HIGH-GAIN FEEDBACK SAMPLED-DATA SYSTEMS VIA SINGULAR PERTURBATION APPROACH

HOSSEIN OLOOMI and MAHMOUD E. SAWAN
EE Dept., Wichita State Univ., Wichita, KS 67208

SUMMARY

Singular perturbation methodology is used to analyze high-gain feedback sampled-data systems. A simplified method for determination of eigenvalues and transmission zeros is developed. A two-stage pole placement design is presented which is shown to be useful when designing deadbeat strategies.

PROBLEM FORMULATION

Consider a sampled-data system

$$x_0(n+1) = (I+\varepsilon A_0)x_0(n) + \varepsilon B_0 u(n) \tag{1}$$

where $x_0 \in R^n$, $u \in R^m$, and the sampling period $\varepsilon \ll 1$. We note that the eigenvalues of (1) are given by

$$\lambda_i(I+\varepsilon A_0) = 1 + \varepsilon\lambda_i(A_0) \quad ; \quad i=1,2,\ldots,n \tag{2}$$

implying that when $\lambda_i(A_0) = O(1)$, eigenvalues of (1) are inside the circle centered at z=1 in the z-plane with radius of $O(\varepsilon)$. It is desired to study (1) under the high-gain feedback control law of the form

$$u(n) = gC_0 x_0(n) \triangleq gy(n) \tag{3}$$

where g is the large scalar gain factor. Motivated by the fact that in designing deadbeat strategies the magnitude of the control signal increases with decrease of the sampling period [1], we let $\varepsilon \triangleq 1/g$ and study the closed-loop system

$$x_0(n+1) = [(I+\varepsilon A_0) + B_0 C_0]x_0(n) . \tag{4}$$

Following work of [2], we assume that $\mathrm{rank}(B_0 C_0) = \mathrm{rank}(C_0 B_0) = m$ and $|\lambda_i(I+B_0 C_0)| < 1$ for $i=1,\ldots,m$. Then the studying of the eigenvalues reveals that (4) possesses the two-time-scale property. Furthermore, defining the slow time-scale k as $\eta=k[1/\varepsilon]$ for $k=0,1,2,\ldots$ and studying the slow variables in the time-scale k shows that the output y(n) reduces to an $O(\varepsilon^2)$ quantity after $n \geq n_s$ steps [3]. Now using the transformation

$$x(n) = [x_1^T(n) \quad x_2^T(n)]^T = Mx_0(n) \quad , \quad MB_0 = \begin{bmatrix} 0 \\ \hline B_2 \end{bmatrix} \tag{5}$$

with $x_1 \in R^{n-m}$, $x_2 \in R^m$, snf B_2 being nonsingular, (4) transforms to

$$x_1(n+1) = (I+\varepsilon A_{11})x_1(n) + \varepsilon A_{12}x_2(n) \tag{6a}$$

$$x_2(n+1) = \varepsilon A_{21}x_1(n) + (I+\varepsilon A_{22})x_2(n) + \varepsilon B_2 u(n) . \tag{6b}$$

We have the following theorem:

Theorem: Let $(I+\varepsilon A_0, \ \varepsilon B_0)$ be a controllable pair, and let the $m\times(n-m)$ matrix K_s and the $m\times m$ matrix K_f be such that

$$\lambda_j[I+\varepsilon(A_{11}-A_{12}K_s)] = 1+\varepsilon q_j \quad ; \quad j=1,\ldots,n-m \tag{7a}$$

$$\lambda_i(I+B_2K_f) = p_i \quad ; \quad |p_i| < 1 \quad ; \quad i=1,\ldots,m \tag{7b}$$

where $(1+\varepsilon q_i)$ and p_i are the prescribed locations of the slow and fast eigenvalues, respectively. Then the use of the high-gain feedback

$$u(n) = gKx(n) = 1/\varepsilon[K_f K_s x_1(n) + K_f x_2(n)] \tag{8}$$

places the eigenvalues of the system (6a), (6b) to $1 + \varepsilon[q_j + O(\varepsilon)]$ and $p_i + O(\varepsilon)$, or equivalently to $1 + 1/g[q_j + O(1/g)]$ and $p_i + O(1/g)$, respectively.

The above theorem suggests that an asymptotic output deadbeat response can be achieved via the high-gain feedback control (8) if the fast eigenvalues are assigned to an $O(\varepsilon)$ neighborhood of the origin. This is possible since the (n-m) remaining slow eigenvalues would automatically be cancelled inside the unit circles with the transmission zeros.

Corollary: Let $(I + \varepsilon A_0, \varepsilon B_0)$ be a controllable pair, and let the $m\times(n-m)$ matrix K_s and the $m\times m$ matrix K_f be such that

$$\lambda_j[I+\varepsilon(A_{11}-A_{12}K_s)] = 1+\varepsilon q_j \quad ; \quad j=1,\dots,n-m \tag{9a}$$

$$\lambda_i(I+B_2K_f) = O(\varepsilon) \quad ; \quad i=1,\dots,m \tag{9b}$$

where $(1+\varepsilon q_j)$ are prescribed locations of the slow eigenvalues inside the unit circle. Then the use of the high-gain feedback (8) places the eigenvalues of the system (6a), (6b) to $1+\varepsilon[q_j+O(\varepsilon)]$ and $O(\varepsilon)$, respectively, i.e., output deadbeat response would be achieved as $\varepsilon\to 0$.

Remark 1: For very small sampling period, the pole-zero cancellation will occur near the unit circle which may not be desirable. In this case the upper bound ε^* of ε for which (4) possesses the two-time-scale property plays an important role in the asymptotic output deadbeat response design.

Remark 2: The above corollary suggests the use of multirate controller which is attractive in practical implementation.

Remark 3: Using results of [2], it can be shown that eigenvalues of (4) take the following form

$$\lambda_i^f = \lambda_i(I+C_0B_0) + O(\varepsilon) \quad ; \quad i=1,\dots,m \tag{10a}$$

$$\lambda_j^s = \lambda_j(I+\varepsilon F_{11}) + O(\varepsilon^2) \quad ; \quad j=1,\dots,n-m \tag{10b}$$

under our previous assumptions and for $0 \leqq \varepsilon \leqq \varepsilon^*$, where

$$MA_0M^{-1} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad ; \quad C_0M^{-1} = [C_1 \quad C_2] \quad ; \quad F_{11} = A_{11} - A_{12}C_2^{-1}C_1 .$$

Then the limit $\lambda_j(I+\varepsilon F_{11})$ of the eigenvalues λ_j^s are the transmission zeros of the open-loop system (1) with the output $y(n) = C_0x_0(n)$.

REFERENCES

[1] K.J. Astrom and B. Wittenmark, Computer Controlled Systems-Theory and Design, Prentice-Hall Information and System Science Series, 1984.

[2] K. Young, P.V. Kokotovic, and V.I. Utkin, "A Singular Perturbation Analysis of High-Gain Feedback Systems," IEEE Trans. Automat. Contr., Vol. AC-22, No. 6, Dec. 1977, pp. 931-938.

[3] B. Litkouhi and H. Khalil, "Multirate and Composite Control of Two-Time-Scale Discrete-Time Systems," IEEE Trans. Automat. Contr., Vol. AC-30, No. 7, 1985, pp. 645-651.

AN EXPERIMENTAL STUDY OF TEST PATTERN AND RESPONSE COMPRESSION TECHNIQUES FOR BIST†

JOHN T. KING
Department of Mathematical and Computer Sciences
Youngstown State University
Youngstown, Ohio 44555

RAMASWAMI DANDAPANI
Department of Electrical Engineering
University of Colorado
Colorado Springs, CO 80907

KEWAL K. SALUJA
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
Madison, WI 53706

ABSTRACT

In this paper we propose a number of algorithms to compress a set of test vectors and output vectors. The main objective being reduction of storage space to store such vectors. The algorithms are tested on a number of randomly generated test sets and found to be quite effective. Algorithms are also applied to two test sets for actual circuits and found to be satisfactory.

1. INTRODUCTION

The development of very large scale integration (VLSI) has made possible the production of extremely complex single chip digital circuits. It would be erroneous to assume these chips, once produced, will always function correctly. Due to physical limits of the materials used and other possible problems that may develop, it is necessary to test such circuits at regular intervals. It may be argued to exhaustively test the circuit. But for a combinational CUT, circuit under test, with n inputs this requires application of 2^n input vectors and this number becomes unwieldy for large n. It is also possible to employ design for testability technicques [1,2,3] but often such techniques adversely effect the performance.

An alternative to exhaustive testing is using a fault model and generating a set of test vectors, T, to adequately test the circuit. Assume T has N vectors. Let O be the set of output vectors the CUT will produce corresponding to T if the circuit is functioning properly. It is then possible to test the circuit using the scheme shown in Figure 1. In this scheme test set T and outputs O are stored in two ROMs. In this simple scheme the dimension of ROM to store T (called Prestored-T) is n*N. The first part of this paper deals with alternative methods to store T. We will then consider methods to store T and O in a single ROM.

---

†This work was in part supported by the National Science Foundation under Grant DCR-8509194 and by the Graduate School Grant-in-aid from the University of Wisconsin-Madison.

2. PRELIMINARIES

When testing the circuit in Figure 1 a test vector, t_i is applied to the CUT. The CUT generates an output w_i, which is compared to the expected output o_i. An error would occur if $o_i \neq w_i$ for any i. Let $T = \{t_1, t_2, \ldots, t_N\}$ and let $t_i = a_0a_1a_2 \ldots a_{n-1}$, $t_j = b_0b_1b_2 \ldots b_{n-1}$, be two vectors in T. Hamming distance between t_i and t_j is defined as

$$H(t_i,t_j) = \sum_{k=0}^{n-1} \theta(a_k,b_k) \text{ where } \theta(a_k,b_k) = \begin{cases} 1 & \text{if } a_k \neq b_k \\ 0 & \text{if } a_k = b_k \end{cases}$$

In [4], a method to reduce the area required to store T on chip was presented. In this method the set T was expanded and ordered into a set $T_1 = \{s_1, s_2, \ldots, s_M\}$. T_1 contained all of set T, however in T_1, $H(s_i, s_{i+1}) = 1$ for any i. This special property of T_1 makes possible a more efficient storage scheme than storing T. Formal description of the method to obtain T_1 from T is given below as Algorithm P1.

```
Algorithm P1;
Input T; Output $T_1$; vector x;
Begin;
x := $t_1$;
place x in $T_1$;
While ($T \not\subseteq T_1$) do
    Begin;
        Choose $t_i$ to be the first vector from $T - T_1$ such that
             $H(x,t_i) < H(x, t_j)$ for all $t_j$ in $T - T_1$.
        If $d = H(x,t_i)$ then add d - 1 vectors into $T_1$ so that adjacent
             pairs have Hamming distance of 1;
        Add $t_i$ to $T_1$;  x := $t_i$;
    End;
End.
```

The design shown in Figure 2 will produce the set T_1 if the register is initialized to the first vector of T_1, and the ROM contains the position of change between successive vectors of T_1. Let M be the size of T_1. The size of the ROM will be R * M, where r is the smallest integer such that $2^r \geq n$. Clearly M determines actual saving in ROM and also M depends on the method used to generate T_1. Algorithm P1 produced savings up to 50% compared to Prestored-T [4].

Although large M may mean longer testing time, yet in practical environment even if Algorithm P1 doubles the test size, the total time spent in testing will be acceptable.

It is also important that the ROM saving due to application of P1 must be greater than the cost of the added hardward. The natural question to ask is: are there other schemes than the one discussed above [4]? We will study this question in this paper.

3. RIGHT CIRCULAR SHIFT METHODS

In the TPG, Test Pattern Generator, of Algorithm P1, r bits are used to store the position of change between successive vectors of T_1. Thus allowing each bit of change to be any of the n positions. One possible way of saving ROM space would be to use less than r bits to represent each position of change. A Right Circular Method, RCM(z), uses $r - z$, $z > 0$, bits to represent the change between adjacent vectors.

Using only r - z bits to represent each change of position presents a problem in 'mobility'. Unlike Algorith P1, which could complement any of the n positions at any time, RCM(z) can not complement an arbitrary position instead it can only complement a position which is between 0 and $2^{r-z}-1$. For storage and generation of the desired test set in RCM methods we shall use the configuration shown in figure 3. It has a shift register (SR) which is initialized to 100 ... 0 state. The ROM contains the number of positions the SR must be shifted (circular shift) to form the next vector. Clearly at any instant SR contains a single 1 and by repetitive shifts any of the n bits in register R can be complemented.

Let POS be the current position of the 1 bit in the SR. Define RANGE(POS,z) = (POS + i) mod n, $0 \le i \le 2^{r-z} - 1$. For application of RCM(z), a given set T needs to be expanded and ordered into a set T_2 such that

(1) adjacent vectors of T_2 differ in only one bit,
(2) if the SR contains 1 in position POS when R contains t_i, then t_i and t_{i+1} differ in position which is in RANGE(POS,z).

In the remainder of this paper, unless otherwise specified, RCM will refer to RCM(1) and RANGE(POS) = RANGE(POS,1). The choice of z=1 was made for it was found to be most effective after some initial experimentation. However, it does not imply that the results presented in this paper can not be improved for other values of z. Also, without effecting the results significantly we assume POS=0 initially [5].

The first RCM method is presented below. In this method first, vectors of T are arranged in increasing Hamming distance order. Additional vectors are then inserted to obtain T_2 using the Algorithm LINK such that T_2 satisfies the necessary conditions.

```
Algorithm P2;
Input T,POS; Output $T_2$; Vector x;
Begin;
    x := $t_1$; Add x to $T_2$;
While (T $\not\subseteq$ $T_2$) do
    Begin;
        choose $t_i$ to be the first vector from T - $T_2$ such that
            $H(x,t_i) \le H(x,t_j)$ for all $t_j$ in T - $T_2$;
        Call LINK(POS,x,$t_i$,$T_2$);
        x := $t_i$;
    End;
End.
```

```
Algorithm LINK;
Input POS,x,y,$T_2$; Output $T_2$;
Begin;
    While (x $\neq$ y) do
        Begin;
            Let q be the next position from POS in which y differs from x
                in a right circular manner:
            If q is in RANGE(POS) then
                Begin;
                $x := a_0 a_1 \ldots \bar{a}_q \ldots a_{n-1}$; {complement position q in x}
                add x to $T_2$; POS := q;
                End;
            Else Begin;
                    $k := (POS + 2^{r-1} - 1) \bmod n$;
                    $x := a_0a_1 \ldots \overline{a}_k \ldots a_{n-1}$;
                    add x to $T_2$;
                    $x := a_0a_1 \ldots \overline{a}_k \ldots a_{n-1}$;
                    add x to $T_2$;
                    POS := k;
                End;
        End;
End.
```

Some properties of the Algorithm LINK are given in [5].

Example 1:

Let $T = \{0000,0011\}$. Applying RCM Algorithm P2 would yield $T_2 = \{0000,0100,0000,0010,0011\}$. Notice in T_2, 0000 appears twice. T_2 and T_1 are ordered sets and duplication of vectors is permissible in them.

It is interesting to note that Algorithm P1 would produce T_1 identical to T_2, but Algorithm P1 would require a $3 * 2 = 6$ bit ROM, and Algorithm P2 would need a $4 * 1 = 4$ bit ROM. Prestored-T needs a $4 * 2 = 8$ bit ROM.

A number of variations of P2 were implemented. However, we present only two additional algorithms here. Algorithm P3 below is an improvement over P2, for after choosing a vector t_i from T the next vector t_j is chosen such that not only it has minimum Hamming distance from t_i but if $H(t_i,t_j) = 1$ then a t_j is chosen which results into a minimum number of additional vectors to be inserted between t_i and t_j in T_2.

```
Algorithm P3;
Input T; Output $T_2$;
x := $t_1$; add x to $T_2$;
While ($T \not\subset T_2$) do
    Begin;
        y := first vector of $T - T_2$;
        For (all $t_i$ in $T - T_2$) do
            Begin;
                If ($H(x,t_i) = 1$ and $H(x,y) = 1$ and the position of
                    difference of x and y is not in RANGE(POS) and
                    position of difference of x and $t_i$ is in
                    RANGE(POS)) then y := $t_i$;
                Else
                    If ($H(x,t_i) < H(x,y)$) then y := $t_i$;
            End;
            Call LINK(POS,x,y,$T_2$),
            x := y;
    End;
End.
```

Algorithm P3 optimizes the selection of vectors with Hamming distance of one. Algorithm P4 below extends this optimization to all Hamming distances. Define $f(x,y,p)$ to be the number of vectors LINK adds to T_2 when connecting x to y with POS = p.

```
Algorithm P4;
Input T; Output $T_2$;
Begin;
x := $t_1$; add x to $T_2$;
While ($T \not\subseteq T_2$) do
    Being;
        y := first element of T - $T_2$;
        For (all $t_i$ in T - $T_2$) do
            Begin;
                If (H(x,y) = H(x,$t_i$)) then
                    If (f(x,$t_i$,POS) < f(x,y,POS)) then y := $t_i$;
                Else
                    If (H(x,$t_i$) < H(x,y)) then y := $t_i$;
            End;
        Call LINK(POS,x,y,$T_2$);
        x := y;
    End;
End.
```

4. EXPERIMENTAL EVALUATION

All the above algorithms, P1, P2, P3 and P4, were tested on random sets of 16 bit vectors, (n = 16), for N = 32 to 1024. Programs implementing these algorithms were run with 15 random test sets for each N. Tables 1 thru 4 contain the averages of these results. Figure 4 shows their relative performance. The outputs of all individual runs and program listings can be found in [5]. The choices of n and N were made largely due to limiation of compute time and ease of program implementation. From these tables it is evident that Algorithm P4 performs consistently better than any of the other algorithms. Further variations of this algorithm were also tried [5].

Size of T	Size of T_1	New ROM Size	Bits Saved	Savings (%)
32	140	559	-47	-9
64	263	1053	-29	-3
128	470	1881	167	8
256	837	3349	747	18
512	1426	5707	2485	30
1024	2486	9944	6440	40

Table 1. Average Results of Algorithm P1 Compared to Prestored-T.

Size of T	Size of T_1	New ROM Size	Bits Saved	Savings (%)
32	160	481	31	6
64	305	914	110	11
128	550	1649	399	20
256	1012	3035	1061	26
512	1860	5581	2611	32
1024	3429	10287	6097	38

Table 2. Average Results of Algorithm P2 Compared to Prestored-T.

Size of T	Size of T_1	New ROM Size	Bits Saved	Savings (%)
32	160	481	31	6
64	305	914	110	11
128	550	1649	399	20
256	1020	3060	1036	25
512	1877	5630	2562	32
1024	3575	10725	5659	35

Table 3. Average Results of Algorithm P3 Compared to Prestored-T.

Size of T	Size of T_1	New ROM Size	Bits Saved	Savings (%)
32	145	435	77	15
64	266	799	225	22
128	487	1461	587	29
256	899	2696	1400	34
512	1644	4931	3261	40
1024	3000	9000	7384	46

Table 4. Average Results of Algorithm P4 Compared to Prestored-T.

5. THE EXISTENCE OF AN OPTIMAL SOLUTION

The objective of the algorithms of section 3 is minimization of the size of set T_2 in accordance with conditions of RCM. This problem can be formualted in terms of finding a minimum weight Hamiltonian path in a graph. Define a graph whose node set consists of elements of T. For each possible value of POS, there is an edge from vector t_i to t_j weighted by $f(t_i,t_j,POS)$. Formally, edges can be labeled $(P^k_{ij},F(i,j,p))$, where P^k_{ij} = the ending value of POS after linking t_i to t_j with POS = k, and $F(i,j,p) = f(t_i,t_j,POS = k)$, i.e., the number of linking vectors. Therefore minimization of T_2 is equivalent to finding a minimum weight Hamiltonian path [6] with the following additional constrain:

If t_i,t_j,t_q are three consecutive vectors in the path, and path from t_i to t_j is labelled $(P^k_{ij}, F(i,j,k))$ then the path chosen from t_j to t_j should be labelled $(P^\alpha_{jq}, F(j,q,\alpha))$ where $\alpha = P^k_{ij}$. It is be strongly believed, though not completely proved, at this time that finding an optimal solution is an NP complete problem.

6. INPUT AND OUTPUT COMPRESSION

All algorithms presented in previous sections ignore the output set O from Figure 1. Compression of the output set is very important. If for every linking vector added to set T_1 or T_2 an additional output vector is added directly to output set O_1, two problems will arise:

(1) O_1's increase in size from set O will negate any ROM savings the compression method produced.

(2) How will the testing circuit be able to differentiate valid test and output vectors from the don't care vectors produced from the compression method implemented?

These problems must be answered if the algorithms presented in this paper are to have any practical value.

A simple way to simultaneously compact Prestored-T and Prestored-O is to combine the two ROMs into a single ROM, V, as shown in Figure 5. ROM V would have N vectors, the same as T and O. A vector v_i from V would be $v_i = t_i o_i$ where t_i is the i^{th} vector from set T and o_i is the corresponding valid output vector from O. Any compaction method we have studied could be used on set V as we did on set T. Let vectors from T be k bits, and vectors from O be s bits. Then vectors from V would be n = k + s bits.

Suppose a RCM was applied to set V producing set V^1. The hardware needed to implement this scheme is almost identical to Figure 3 with a few additions as illustrated in Figure 6. To distinguish valid output from the invalid output produced by the RCM employed, a straight forward approach is to place an extra bit on the end of each vector in V^1. Let x_i be a vector of V^1. If $x_i = v_k$ for some v_k in set V then the extra bit would be 1, otherwise the bit would be 0. This extra bit is ANDed with the output of XOR2. Our results were promising only for large test sets and performed poorly for small test sets. An alternative approach proposed in [7] was not implemented. In any case we believe that a thorough analysis of input and output compression techniques still remains to be done.

7. CONCLUSIONS

In this paper we addressed the problem of test vector and output vector storage. A class of algorithms, RCM, as an alternative to Prestored-T and Prestored-O is proposed and studied in detail. Our experimental study on random data shows that the Algorithm P4 performs better than any of the known algorithms. These algorithsm were also applied to an actual test set, (although for a small circuit) and the results of these are given in Appendix 1.

One method for combining Prestored-T and Prestored-O is also discussed but further work needs to be done in this direction.

Although, we have presented the results in a manner indicating their suitability only in BIST environment. This is not completely true for the scheme proposed in this paper can also be used for storage of test vectors in the external tester environment. The method is likely to reduce the storage space considerably at a little additional cost of increase in the complexity of a test program.

REFERENCES

1. T. W. Williams and K. P. Parker, "Design for Testability - A Survey," Proc. IEEE, Vol. 71, No. 1, January 1983, pp. 98-112.

2. E. J. McCluskey, "Verification Testing - A Pseudoexhaustive Test Technique," IEEE Trans. Computers, Vol. C-33, No. 6, June 1984, pp. 544-546.

3. C. R. Kime, H. H. Kwan, J. K. Lemke and G. B. Williams, "A Built-In Test Methodology for VLSI Data Paths," Digest, 1984 IEEE Test Conference, October 1984, pp. 327-337.

4. R. Dandapani, J. H. Patel and J. A. Abraham, "Design of Test Pattern Generators for Built-In Test," Digest, 1984 IEEE Test Conference, October 1984, pp. 315-319.

5. J. T. King, An Experimental Study of Test Pattern and Response Compression Techniques for BIST, M.S. Thesis, Dept. of Mathematical and Computer Sciences, Youngstown State University, Ohio, August 1986.

6. F. Harary, Graph Theory, Addison-Wesley, 1969.

7. P. Sheen, R. Dandapani, K. K. Saluja, "A Test Pattern Generator with Embedded Compressed Output for Built-In Self-Test," Proceedings of Twenty Third Annual Allerton Conference on Communication, Control and Computing, October 1985, pp. 202-203.

APPENDIX 1

Results with Actual Test Sets

The following two tables give the results of Algorithms P1-P4 on actual test sets. The two test sets consist of 27 and 46 test vectors with n = 14.

Algorithm	Size of T2	New ROM SIZE	Bits Saved	Savings (%)
P1	79	316	62	17
P2	95	285	93	25
P3	106	318	60	16
P4	93	279	99	27

Algorithms P1-P4 applied to an actual Test set, N = 27, n = 14.

Algorithm	Size of T2	New ROM SIZE	Bits Saved	Savings (%)
P1	124	496	148	23
P2	152	456	188	30
P3	162	486	158	25
P4	162	486	158	25

Algorithms P1-P7 applied to an actual Test set, N = 46, n = 14.

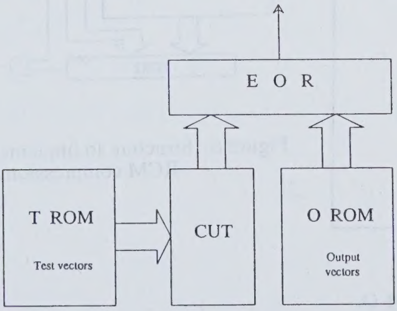


Figure 1: Prestored-T and Prestored-O for BIST.

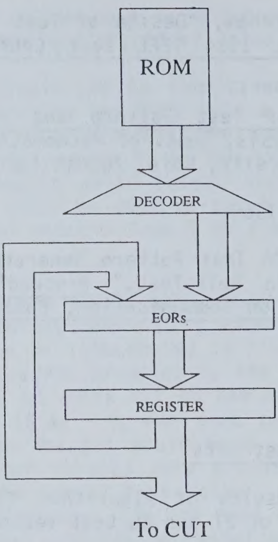

ROM
DECODER
EORs
REGISTER
To CUT

Figure 2: Structure to implement Algorithm P1.

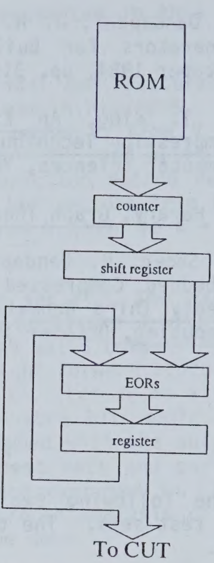

ROM
counter
shift register
EORs
register
To CUT

Figure 3: Structure to implement RCM.

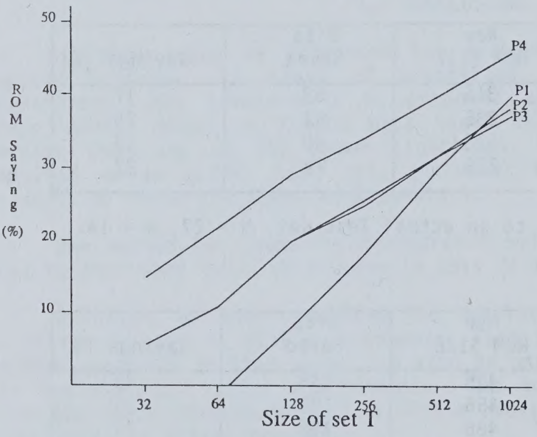


Figure 4: Average performance of Algorithms P1-P4.

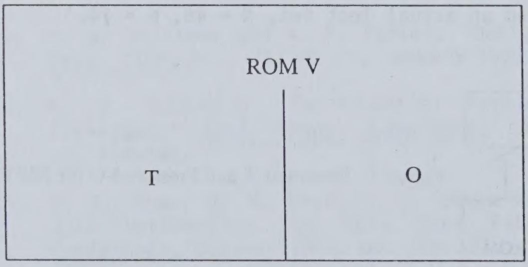


Figure 5: Combined Prestored-T and Prestored-O.

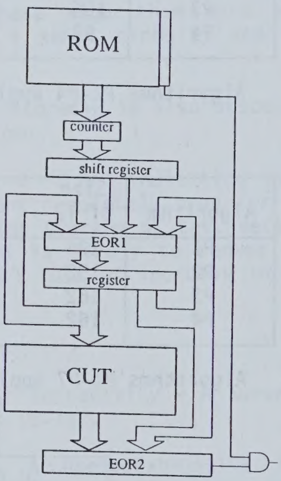

ROM
counter
shift register
EOR1
register
CUT
EOR2

Figure 6: Structure to implement RCM compression of V.

STV-TESTABILITY: AN APPROACH TO TESTING BILATERAL SYSTOLIC ARRAYS

HASAN ELHUNI
School of Engineering
University of Portland
Portland, OR 97203-5798

ANASTASIOS VERGIS
Dept. of Computer Science
University of Minnesota
Minneapolis, MN 55455

ABSTRACT

An approach called STV-testability is introduced in this paper for testing bilateral systolic arrays. In bilateral systolic arrays, information flows in opposite directions. STV-testability facilitates testing such arrays in three phases: Set-Test-Verify. Using STV-testability, bilateral systolic arrays can be tested using a minimal number of test vectors, which is the number required to test a single cell. The test set is easy to generate and verify, which makes STV-testability attractive for Built-In-Self-Testing (BIST). Bilateral systolic arrays which are used for matrix-vector multiplication are used to illustrate the approach.

1.0 INTRODUCTION

In recent years, many systolic systems have been designed for complex computations such as in the areas of digital signal processing and artificial intelligence. Several systolic arrays are available for matrix and convolution operations [KUNG-82]. In a systolic system, data flows from the computer memory in a rhythmic fashion, passing through many processing elements before it returns to memory. Systolic arrays have many advantages such as the relatively low design cost because of their regular structure, and the possibility of expanding the system to acheive higher performance. Moreover, graceful degradation of the processing power in the event of failures in some of the system modules is another desirable feature of systolic arrays.

An important problem associated with such systems is fault detection; that is, derivation of test input sequences which can be applied to the array so that the output sequences of the normal and any faulty array (under some fault assumptions) are different. In this paper, we introduce an approach called STV-testability (Set-Test-Verify) for testing such arrays.

2.0 STV-TESTABILITY

The issue of testing bilateral arrays of combinational cells was discussed before in [GRAY-78] and [VERG-86]. Test procedures were suggested to test such arrays with a linear test set; that is, the number of test vectors changes linearly with the number of cells of the array. The test procedure can be completed also in a linear number of time units. The basic cell of bilateral systolic arrays receives three input signals (x,y,z) and it generates two output signals (x^0,z^0) as illustrated in Figures 1 and 2. A cell with input (x,y,z) has a fault that causes the error $(x^0/x',z^0/z')$ if the cell generates (x',z') such that $(x^0,z^0)\neq(x',z')$. We assume the Single Cell Fault

Model (SCFM) and combinational cells with registers to hold the intermediate results. The next theorem gives sufficient conditions for STV-testability.

Theorem 1: A bilateral systolic array is STV-testable if there exists an input $y_a \in Y$ that propagates the signals $x \in X$ and $z \in Z$ unchanged, i.e.,

1- $x^o(x,y_a,z) = x$ independent of z.

2- $z^o(x,y_a,z) = z$ independent of x.

Proof: Suppose that the input (x,y,z) is one of the inputs required to test a single cell in a bilateral systolic array. By using the input y_a, the x and z inputs can be shifted through the array in (N-1) time units if the array has N cells. This is the Set-up phase of STV-testability. Then, the input y is applied to all cells simultaneously in one time unit to test all cells for input (x,y,z). The output signals are shifted to the observable outputs for verification using y_a for (N-1) time units. It can be noticed here that we test all cells simultaneously for a certain input in one time unit. This is different from the approaches in [GRAY-78] and [VERG-86], where each cell is tested completely starting from the rightmost cell. This causes the difference between the linear number of test vectors in [GRAY-78] and [VERG-86], and the minimal (constant) number of test vectors using STV-testability.

Another important issue associated with testing systolic arrays is the testing time. The testing time of a systolic array is the number of time units required to completely test the array. The next theorem addresses this issue.

Theorem 2: An STV-testable bilateral systolic array of N cells can be tested with a minimal number of test vectors and in a linear number of time units.

Proof: The testing procedure consists of three parts. The first one is to set-up the array using the special input y_a. In the second part, which is the test part, all cells are tested for the same input. The third part is to shift all outputs using y_a for verification. The actual number of test vectors to test the entire array is equal to the number of test vectors required to test a single cell, which is the minimal number of test vectors. Each test vector requires (N-1) time units for set up, one unit to test, and (N-1) units to verify. The set up and verification procedures can be overlapped by (N-1) time units for consecutive test vectors. If p test vectors are required to test a single cell, then (N-1)+pN time units are required to test the whole array.

3.0 MATRIX-VECTOR MULTIPLICATION

Several systolic arrays have cells that perform simple operations such as the additive-inner product. In bilateral systolic arrays for matrix-vector multiplication, the basic cell receives the inputs (x,y,z), and it generates the outputs $x^o=x$ and $z^o=z+xy$ as demonstrated in Figure 3. The basic cell consists of an adder and a multiplier. Methods to test adders and multipliers are discussed in [SRID-81] and [SHEN-84], respectively. Here, we assume that the input (a,b,c) is one of the inputs required to test a single cell. The testing procedure is demonstrated in Figure 4 for a bilateral systolic array of four cells. To set up the array, the input $y_a=0$ is used for (N-1) time units to propagate a and c to all cells. The vector bbbb is applied to test all cells for the input (a,b,c). To propagate the outputs a^o and c^o, the inputs $y_a=0$ is used again for (N-1) time units.

4.0 BUILT-IN SELF TEST

Built-In-Self-Test (BIST) refers to the idea of testing a system with little interaction with the outside world [PATE-84]. The test generator and the test verifier are included within the system under test as shown in Figure 5. A BIST for a chip is built on the same chip, while a BIST for a board is built on the same board. BIST consists of two major parts:

1- Test Generation: STV-testable arrays have simple test generation procedure. The test vectors for the entire array are repetitions of the test inputs for a single cell. Therefore, The test set for a single cell can be stored in a ROM and used for the entire array.

2- Test Verification: Cells in STV-testable arrays are tested for the same input at the same time. Then, the output responses of all cells are shifted to the array boundaries. These output responses are identical; therefore, they can be compared with one another. The output signals at the boundary cells at time t are compared with the output signals at time t-1. This can be done by storing the outputs at the boundaries for one time unit using a register until they are compared with the outputs at the next time unit. The problem with this approach is that it does not detect faults that cause a valid test to be executed other than the intended one. In this case, outputs from all cells will be identical but incorrect. Another approach is to store the output responses of a single cell in a ROM. Using this technique, the test generation ROM and the response verification ROM check each other for correct operation.

5.0 DISCUSSION

In this paper, we presented an approach called STV-testability for testing bilateral systolic arrays. This approach has many advantages; the test set is easy to generate because of its similarity to the test set for a single cell. The number of test vectors is minimal and the number of time units required to completely test the array is linear. The test verification procedure is also simple because all cells produce identical responses and the number of output responses is minimal. This makes STV-testable arrays suitable for Built-In-Self-Test. STV-testability approach can be applied to many existing systolic arrays such as bilateral arrays for matrix-vector multiplication.

REFERENCES

[ELHU-86] H. Elhuni, "Techniques for Design and Testing of Iterative and Systolic Arrays", Ph.D. Thesis, Elect. Eng. Dept., Univ. of Minnesota, 1986.

[GRAY-78] F. G. Gray and R. A. Thompson, "Fault Detection in Bilateral Arrays of Combinational Cells", IEEE Trans. Comput., Vol. C-27, pp. 1206-1213, Dec. 1978.

[HWAN-84] K. Hwang and F. Briggs, Computer Architecture and Parallel Processing, McGraw-Hill, 1984.

[KUNG-82] H. T. Kung, "Why Systolic Architectures?", Computer, Jan. 1982, pp. 37-46.

[MEAD-80] C. Mead and L. Conway, Introduction to VLSI systems, Addison-Wesley, 1980.

[PATE-84] J. H. Patel, "Built-In-Test for Bit-Sliced ALU", Int. Conf. on Computer Design, 1984.

[SHEN-84] J. P. Shen and F. J. Ferguson, "The Design of Easily Testable VLSI Array Multipliers", IEEE Trans. Comput., Vol. C-33, pp. 554-560, June 1984.

[SRID-81] T. Sridhar and J. Hayes, "A Functional Approach to Testing Bit Sliced Microprocessors", IEEE Trans. Comput., Vol. C-30, pp. 563-571, Aug. 1981

[VERG-86] A. Vergis and K. Steiglitz, "Testability Conditions for Bilateral Arrays of Combinational Cells", IEEE Trans. Comput., Vol. C-35, pp. 13-22, Jan. 1986.

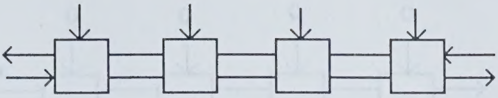


Figure 1. Bilateral systolic array.

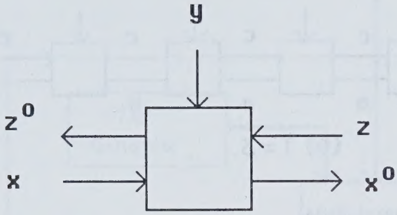


Figure 2. The basic cell.

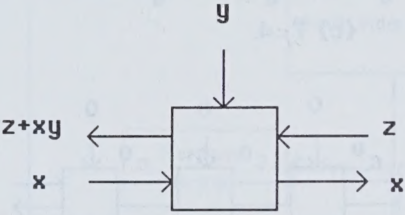
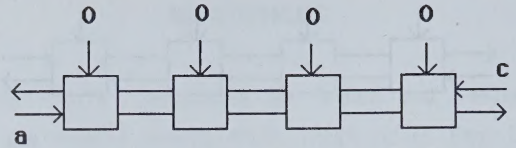
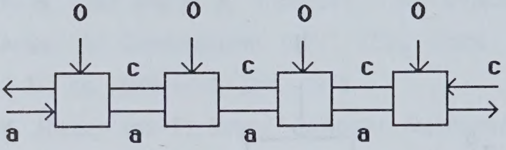


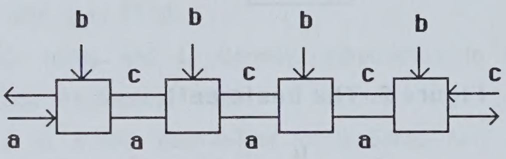
Figure 3. Matrix-vector multiplication



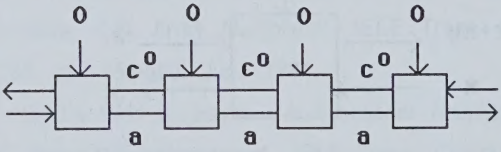
(a) T=0.


0
0
0
0
c
c
c
c
a
a
a
a

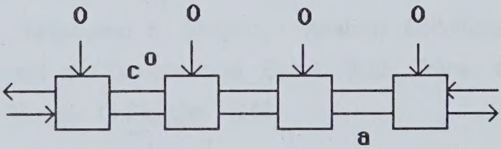
(b) T= 3.


b
b
b
b
c
c
c
c
a
a
a
a

(c) T=4.


0
0
0
0
c0
c0
c0
a
a
a

(d) T=5.



(e) T=7.

Figure 4. Testing a bilateral systolic array.

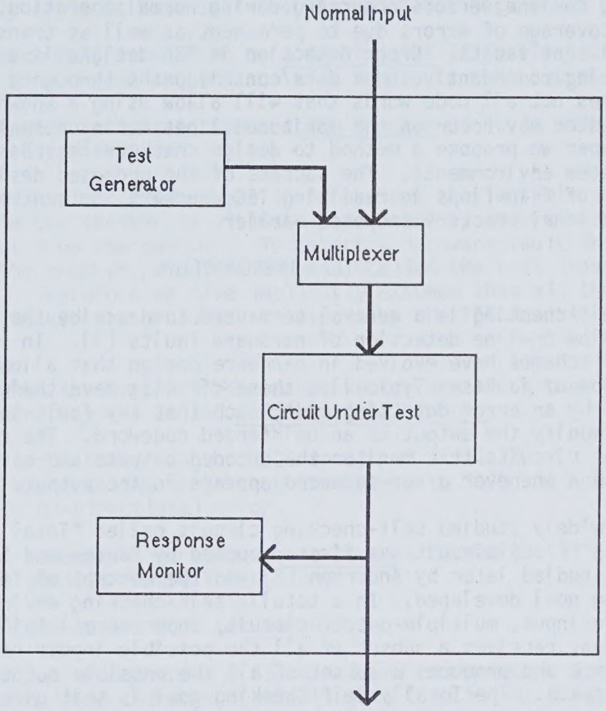

Normal Input
Test Generator
Multiplexer
Circuit Under Test
Response Monitor

Figure 5. A general organization for built-in-test.

ON THE DESIGN OF EMBEDDED TSC CHECKERS*

SANDIP KUNDU AND SUDHAKAR M. REDDY
Department of Electrical and Computer Engineering, University of Iowa, Iowa City, Iowa 52242

ABSTRACT

Totally-Self-Checking (TSC) circuits are used in computer systems to detect, on-line, errors occurring during normal operation. TSC circuits allow coverage of errors due to permanent as well as transient and intermittent faults. Error detection in TSC designs is achieved by monitoring redundantly coded data/control paths through a TSC checker. Sometimes not all code words that will allow using a known design for a TSC checker may occur on the monitored lines during normal operation. In this paper we propose a method to design checkers that is applicable to all system environments. The success of the proposed design is due to the use of flip-flops in realizing TSC checkers, in contrast to mostly combinational checkers proposed earlier.

I. INTRODUCTION

Self-checking is a general term used to describe the logic networks that allow on-line detection of hardware faults [1]. In recent years, several schemes have evolved in hardware design that allow on-line detection of faults. Typically, these circuits have their outputs encoded by an error detecting code, such that any fault in the circuit cannot modify the output to an unintended codeword. The checkers are a class of circuits that monitor the encoded outputs and give an error indication whenever a non-codeword appears in the output.

A widely studied self-checking circuit called "Totally Self-Checking" (TSC) circuit was first proposed by Carter and Schneider [2]. It was studied later by Anderson [3] and the concept of Totally Self-Checking goal developed. In a totally self-checking environment a multiple-input, multiple-output circuit, under normal failure free operation, receives a subset of all the possible inputs called the input code space and produces a subset of all the possible outputs called the output space. The Totally Self-Checking goal is that given a faulty situation a non-code space output is the first erroneous output. Naturally, the checker that monitors the output of a TSC circuit should also be Totally Self-Checking. In other words a checker for a TSC circuit should give an error indication whenever it receives a non-code output from the functional unit or whenever there is a hardware fault in itself. In our design, we provide coverage for single faults. Definitions given below are from [1-5].

---

*The research reported was supported in part by U.S. Army Research Office Grant No. DAAG-29-84-K-0044.

Definition 1: A circuit is called Fault-Secure for a fault-set F, if and only if for any $f \epsilon F$ the circuit never produces an incorrect code space output for code space inputs.

Definition 2: A circuit is called Self-Testing for a fault-set F if and only if for any $f \epsilon F$ the circuit produces a non-code space output for at least one code space input.

Definition 3: A circuit is said to be Totally Self-Checking for a fault-set F if and only if it is both Self-Testing and Fault-Secure for F.

Definition 4: A circuit is said to be Code-Disjoint if it always maps non-code inputs to non-code outputs and code inputs to code outputs.

Definition 5: A circuit is said to be a TSC Checker if it is both self-testing and code-disjoint.

From the definitions given above it is obvious that a fault in the TSC circuit or in the checker is ultimately detected by observing a non-code space output from the checker. To detect a hardware fault in the TSC circuit or the checker, specific input(s) called the test input(s) must be applied. Therefore we have implicitly assumed that all the test-inputs for the checker are produced as outputs by the TSC functional unit being monitored.

Definition 6: When the errors in an output vector are all either 1-to-0 or all 0 to 1 then they are called unidirectional errors.

Example 1: The error that changes 11001 vector to 11111 is unidirectional, where as the error that changes it to 10101 is a non-unidirectional or bi-directional error.

Definition 7: A codeword X is said to one-cover a codeword Y if it has a one in every bit position where Y has a one.

Example 2: 1001 one-covers 1000 but not 0101.

Definition 8: A set of codewords I is said to be unordered if for any codeword $X \epsilon I$ there is no codeword Y belonging to I such that X one-covers Y or Y one-covers X.

If unidirectional errors at the output of a TSC functional unit are to be detected by a TSC checker then these outputs must be encoded in an unordered code [3,4,6].

The output of a TSC checker should also belong to an eror detecting code. The simplest such codes are dual rail codes. If the output of a checker belongs to the set {10,01} we assume an error-free situation. The output of a TSC checker becomes 00 or 11 if and only if an error has been detected in the output of the functional unit or there has been a hardware fault in the checker itself. The fault model considered in this paper includes line stuck-at-zero and line stuck-at-one faults.

It has already been mentioned that a Totally Self-Checking checker must be code-disjoint. Often we come across situations where if a checker is designed for code-disjointness, all the test inputs it needs for testing faults are not available from the TSC functional unit. Thus, though this checker is code-disjoint it is not self-testing. In order to make it self-testing we may need to suspend normal operation and apply the test inputs not available from the functional unit [5,7]. Our objective in this paper is to design checkers which require no external access and can be easily embedded, yet have both the self-testing and code-disjoint property.

In the following section we briefly review some of the concepts applied to our design. Section III includes the proposed checker design. An application of the proposed TSC checker design is discussed in Section IV. Concluding remarks are given in Section V.

II. PRELIMINARIES

A block diagram of a TSC system is shown in Figure 1. The check bits (cf. Figure 1) are normally chosen such that the functional block outputs form an unordered code. In the design we propose, we would additionally require that the check bits be selected to allow the design of TSC checkers. As explained in the next section, this may require partitioning the set of functional outputs such that the bits in each block of the partition can be coded into an unordered code for which a TSC checker with two outputs can be designed. The TSC checker proposed in the next section uses a tree network of two-input two-rail comparators given earlier in [3]. Inputs to a two input two-rail comparator are made up of two pairs of inputs $[(a_1,a_2), (b_1,b_2)]$ where $b_i = \bar{a}_i$. The two outputs f,g of the two input comparator are defined below and a circuit realization of it together with the four required tests for it are given in Figure 2.

$f = a_1b_2 + a_2b_1$

$g = a_1a_2 + b_1b_2$

As mentioned in Section I, most proposed TSC checkers for error-detecting codes have two outputs that produce either 01 or 10 output in normal operation. Hence it may be possible to monitor the outputs of 2 TSC checkers by using a two-input two-rail checker as shown in Figure 3. The overall circuit of Figure 3 would be TSC if and only if f_1g_1, f_2g_2 provide all the four tests given in Figure 2(b). For example if f_1g_1, f_2g_2 would only take on the two values given in Figure 4, clearly the checker of Figure 3 is not TSC. In this case all tests for the two-rail comparator can be generated by inserting clocked delay (D) flip-flops in the outputs of one of the TSC checkers as shown in Figure 5 (the insertion D flip-flops can be in the output of either of the first level checkers). It can be readily verified that all tests for the two rail comparator are now available. The requirement of clocked D flip-flops in the design of Figure 5 checker does not pose difficulty since in most digital systems clocked flip-flops are natural parts. In the next section we give an algorithm that will lead to TSC checkers for any functional block.

III. PROPOSED DESIGN

Let $y_1, y_2, \ldots, y_k$ be the functional outputs (cf. Figure 1). The procedure given below leads to TSC functional unit and a TSC checker.

PROCEDURE TSC CHECKER DESIGN

1. Partition $y_1, y_2, \ldots, y_k$ into blocks such that the bits in each block can be encoded into an unordered code for which a normal two output TSC checker exists.

2. Design two output TSC checkers for the codes chosen in Step 1.

3. Monitor the outputs of the TSC checkers through a tree of two input two-rail comparators. When necessary use delay flip-flops on a pair of inputs of the TSC two rail comparator as shown in Section II.

An example given below illustrates the design procedure outlined above.

Example 3: Let the expected output space of the functional unit (information part of the coded outputs) be as given below. Note that only 8 of the possible 32 tuples appear at the functional unit outputs.

Input #	y_1	y_2	y_3	y_4	y_5
1	0	0	0	1	0
2	0	1	0	0	1
3	0	0	1	1	0
4	1	1	0	0	1
5	1	0	1	0	1
6	0	1	1	0	1
7	1	1	1	1	0
8	1	0	0	0	1

Now we apply our algorithm to show how a TSC checker can be designed for this output table. Partition the outputs as $\{y_1, y_2, y_3\}$ and $\{y_4, y_5\}$. Encode each partition as an unordered code. Note that $\{y_4, y_5\}$ is already unordered and hence no encoding is required for y_4 and y_5. $\{y_1, y_2, y_3\}$ can be encoded into a Berger code, by using two check bits d_1 and 2. Thus the resultant encoded output space is:

Input #	y_1	y_2	y_3	d_1	d_2	y_4	y_5
1	0	0	0	1	1	1	0
2	0	1	0	1	0	0	1
3	0	0	1	1	0	1	0
4	1	1	0	0	1	0	1
5	1	0	1	0	1	0	1
6	0	1	1	0	1	0	1
7	1	1	1	0	0	1	0
8	1	0	0	1	0	0	1

A TSC checker for the Berger code part can be constructed from the results of [10]. Referring to Figure 6 one sees that the output of the TSC checker for the Berger code and $y_4 y_5$ have the following values.

Input #	O_1	O_2	y_4	y_5
1,7	1	0	1	0
2,4,5,6,8	0	1	0	1
3	0	1	1	0

Hence a two-rail comparator to monitor these outputs requires one of the input pair delayed as shown in Figure 6.

For the example studied above the procedure given earlier in [7] to design TSC systems would also require a minimum of two check bits. However it is known that this procedure may not always lead to TSC checkers [7]. It can be shown that Example 3 is one such case.

To prove that the proposed method is applicable to all systems, we need to show that given any functional outputs $y_1, y_2, \ldots, y_k$ it is possible to apply Steps 1 and 2. Additionally we need to show that given any arbitrary pair of two bit inputs $\{(a_1,b_1), (a_2,b_2)\}$ such that $a_i = \bar{b}_i$, $i = 1,2$, it is possible to construct a TSC two input two-rail comparator to monitor (a_1,b_1,a_2,b_2). The first requirement is trivially satisfied by encoding each y_i into a two-rail code by adding one check bit $\bar{y}_i$ for each y_i. The second requirement given above can readily be shown to be satisfied if none of the functional outputs is a constant (i.e. some y_i is not 0 or 1 all the time). Proofs of these claims are omitted, as they are straightforward but unrewardingly long.

IV. AN APPLICATION: TSC CHECKERS FOR AN CODES

Design of TSC checkers for AN codes were given by Gaitanis [8], when $A = 2^m - 1$, where m is an integer. TSC checkers for AN codes are not known when A is arbitrary. Suppose N requires n binary bits for representation and A requires m binary bits. We can apply our design to show that a TSC checker can always be designed for any AN code, when A is not divisible by 2^ℓ. Note that when $A = 2^\ell$, AN codes are not error detecting and when $A = r2^\ell$ then it tantamounts to generating a code by $A = \gamma$ and then augmenting it by ℓ number of zero bits which does not enhance the ability of the code. When we multiply a m bit number (N) with a n bit number (A) we obtain a m+n bit word. In our design we select the n LSBs of such a codeword and use them to generate the complements of m MSBs of the codeword. A TSC dual rail checker which may employ delays checks for agreement between the m MSBs of the codeword and the generated ones (cf. Figure 7). To ensure self-testing property of the MSB regenerator block we must ensure that all combinations are available for the last n bits. Suppose N_1 and N_2 are two different numbers. If $AN_1 \bmod 2^n = AN_2 \bmod 2^n$ then $(AN_1-AN_2) \bmod 2^n = 0$ which implies that 2^n divides A. We do not consider AN codes when A is divisible by 2^n for reasons stated before. Thus two numbers N_1, N_2 $(N_1 \neq N_2)$ maps into two different values for the last n bits of a m+n length AN code. Since there is a 1 to 1 mapping, all combinations of the n LSB's of this m+n length code are available.

For testability of the dual-rail checker we should ensure that each of the m MSBs appear as zeroes as well as ones in codewords. In other words none of these m bits are constant. This is automatically insured since if any bit of the codeword is known to be a constant, it can be deleted without reducing its error detection capability.

The TSC checker design for AN codes illustrates that the procedures given in this paper may often not require more check bits to achieve TSC systems, than those required for obtaining the desired error detection properties.

V. CONCLUSIONS

In this paper we have given a procedure to design Totally-Self-Checking digital systems. The methods proposed are applicable for all system requirements. The generality of the applicability of the results is mainly due to the use of flip-flops in appropriate places in the design of TSC checkers, which is in contrast to the earlier known designs that mostly used combinational checkers.

REFERENCES

1. J.E. Smith and Gernot Metze, "Strongly fault-secure logic networks", IEEE Trans. on Comp., Vol. C-27, pp. 491-499, June 1978.

2. W.C. Carter and P.R. Schneider, "Design of dynamically checked computers", IFIP 68, Vol. 2, Edinburgh, Scotland, pp. 878-883, August 1968.

3. D.A. Anderson, "Design of self-checking digital networks using coding techniques", Technical Report R-524, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, 1971.

4. J.E. Smith, "Design of totally self-checking combinational logic circuits", Ph.D. Thesis, Report R-737, University of Illinois, Urbana, Illinois, August 1976.

5. S.L. Wang and A. Avizienis, "The design of totally self-checking circuits using programmable logic arrays", Fault Tolerant Comp. Symp., pp. 173-180, June 20-22, 1979.

6. H.J. Beuscher and W.N. Toy, "Check schemes for integrated microprogrammed control and data transfer circuits", IEEE Trans. on Comp., Vol. C-19, pp. 1153-1159, December 1970.

7. G.P. Mak, J.A. Abraham and E.S. Davidson, "The design of PLA's with concurrent error detection", Fault Tolerant Comp. Symp., pp. 303-310, June 22-24, 1982.

8. N.K. Jha and J.A. Abraham, "The design of totally self-checking embedded checkers", Fault Tolerant Comp. Symp., pp. 265-270, June 20-22, 1984.

9. N. Gaitanis, "Totally self-checking checkers for low cost arithmetic codes", Fault Tolerant Comp. Symp., pp. 260-264, June 20-22, 1984.

10. M.J. Ashjaee and S.M. Reddy, "On totally self-checking checkers for separable codes", IEEE Trans. on Comp., Vol. C-26, pp. 737-744, August 1977.

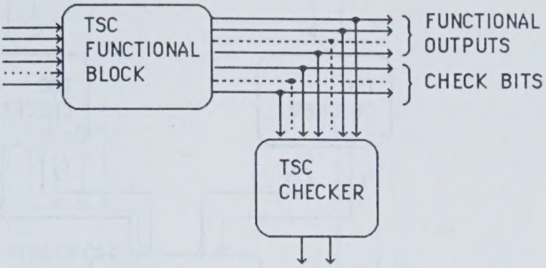

TSC
FUNCTIONAL
BLOCK
FUNCTIONAL
OUTPUTS
CHECK BITS
TSC
CHECKER

Figure 1: A TSC System

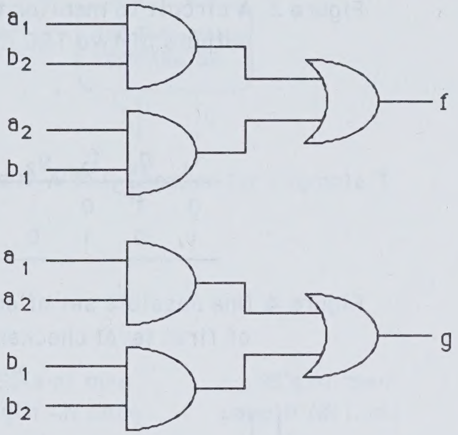

a_1
b_2
a_2
b_1
f
a_1
a_2
b_1
b_2
g

Figure 2(a) : Two-input Two-rail Comparator.

a_1	b_1	a_2	b_2
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0

Figure 2(b) : Tests for the Circuit of Figure 2(a)

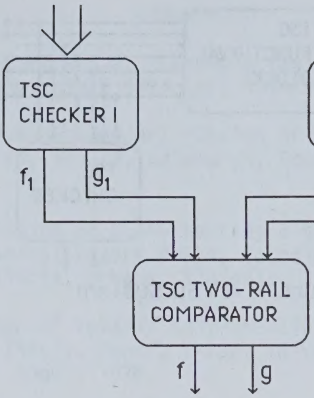

TSC
CHECKER I
TSC
CHECKER II
f_1
g_1
f_2
g_2
TSC TWO-RAIL
COMPARATOR
f
g

Figure 3: A circuit to monitor the outputs of two TSC Checkers.

f_1	g_1	f_2	g_2
0	1	0	1
1	0	1	0

Figure 4: One possible set of outputs of first level checkers in Figure 3.

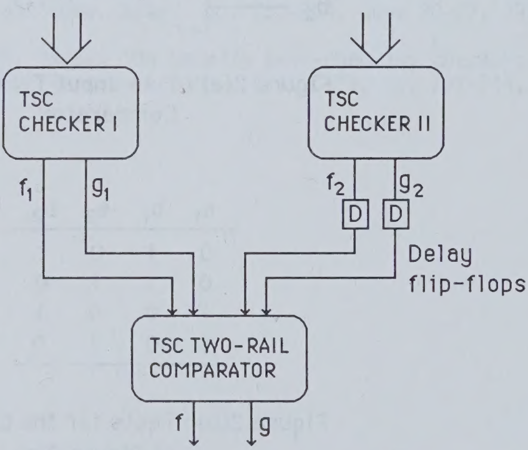

TSC
CHECKER I
TSC
CHECKER II
f_1
g_1
f_2
g_2
D
D
Delay
flip-flops
TSC TWO-RAIL
COMPARATOR
f
g

Figure 5: A TSC Checker for the case of Figure 4 outputs.

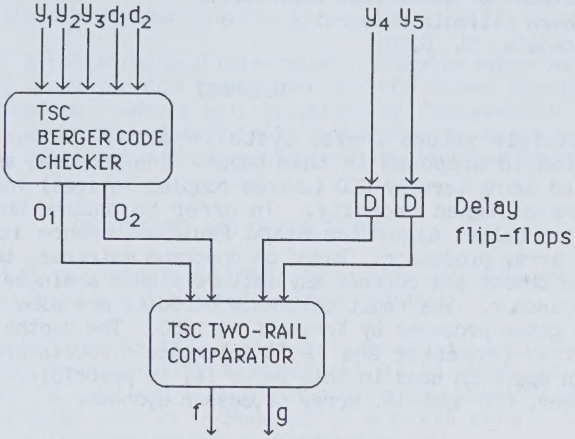

$y_1 y_2 y_3 d_1 d_2$
y_4 y_5
TSC BERGER CODE CHECKER
O_1
O_2
D
D
Delay flip-flops
TSC TWO-RAIL COMPARATOR
f
g

Figure 6: A TSC Checker for Example 3

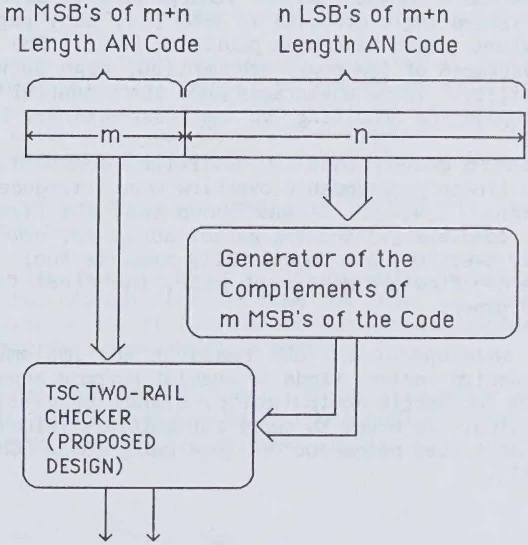

m MSB's of m+n Length AN Code
n LSB's of m+n Length AN Code
m
n
Generator of the Complements of m MSB's of the Code
TSC TWO-RAIL CHECKER (PROPOSED DESIGN)

Figure 7: A TSC Checker for AN Codes.

A FAULT-TOLERANT TERNARY CCD SYSTOLIC ARRAY PROCESSOR

JIA-YUAN HAN

Department of Electrical Engineering
Southern Illinois University
Carbondale, IL 62901

ABSTRACT

A multiple-valued logic, systolic array processor for matrix computation is proposed in this paper. The proposed array processor is constructed using ternary CCD (charge-coupled devices) inner product step processors designed recently. In order to ensure the results highly reliable, so-called algorithm-based fault tolerance is applied to the proposed array processor. Based on checksum matrices, the fault tolerant system can detect and correct any failure within a single processor in the array processor. The fault tolerance circuits are also designed by using basic CCD gates proposed by Kerkhoff in 1980. The depths and cost factors of the array processor and fault tolerance circuits are also discussed. The design approach used in this paper is, in principle, applicable to any radix number, CCD systolic array processor systems.

INTRODUCTION

Multiple-valued logic (MVL) has shown great interest for theoretical research and industrial applications, especially in VLSI technology. Multiple-valued variables, which contain more information than binary variables, can be transmitted among devices with less interconnections. This advantage can reduce interconnection complexity and reduce chip area on VLSI [1,2].

Since Kerkhoff showed that CCD (charge-coupled device) can be used to implement 4-valued logic circuits in 1980 [3], many papers in CCD-based multiple-valued logic have been published [4-7]. The CCD technology has inherent advantages of low power consumption, high packing density, and MOS compatibility. These advantages show its potential in VLSI technology and in development of combining MVL and today's binary logic.

Five basic gates, that is, addition, constant, fixed overflow, inhibit, and linear programmable overflow were introduced by Kerkhoff and his colleague [3,4,8]. It was shown that the first four gates are functionally complete [7] and the gates: addition, constant, and linear programmable overflow, are functionally complete too. Because the linear programmable overflow are most cost gate, the first four gates will be used in this paper.

Recent development in VLSI research and implementation makes it possible to design various kinds of special purpose array processors, such as processors for matrix manipulation, signal processing, and FFT in a VLSI chip [9,10]. In order to carry out multiple-valued array processors, an inner product step processor designed using basic CCD has recently been developed [11].

In addition to achieving high performance in VLSI technology, high reliability is also very important to ensure that the results of long computations are correct. Recently, Abraham and his colleagues [12,13] proposed a system-level fault tolerance method, called algorithm-based fault tolerance, which can detect and correct any failure within a single processor in a multiple processor system.

In this paper, a reliable, multiple-valued, systolic array processor is proposed based on basic CCD gates and the CCD inner product step processor designed before. Matrix multiplication is implemented in this array processor. Algorithm-based fault tolerance is applied to the processor and error detection and correction are implemented using basic CCD gates.

BACKGROUND AND DEFINITIONS

Basic CCD gates and its functions, depth of a CCD realization, and cost of a CCD realization are discussed in this section.

Basic CCD gates used in this paper are constant, addition, fixed overflow, and inhibit introduced by Kerkhoff [3,8] as shown in Fig. 1. Because that delay is used very frequently in systolic array and fault tolerance design, a delay gate is introduced as a basic gate. In fact, the delay gate can be implemented using an addition gate.

The constant gate is a charge generation gate used to convert an input voltage into a discrete charge packet. The output charge of an addition gate is the analog sum of the input charge packets. The fixed overflow gate is used to distribute an input quantity of charge over a number of fixed-capacity storage wells. In this gate, if the input charge exceeds the capacity of the first well, the excess charge is transferred to the second well and so on. The inhibit gate controls the routing of an input charge by means of a control input C. The inhibit gate is implemented based on the floating-gate structure of CCD circuits [4].

The delay between the input and output charge of all basic gates is always one clock period [8]. The output of the delay gate is just the input signal in the previous clock. In order to get different clock-periods delay, many delay gates can be cascaded together. A simple symbol is introduced to express the cascaded delay gates of n as shown in Fig. 2.

The definitions of the depth and cost given in [7] are used in this paper and are restated here.

Definition 1. The depth (D) of a CCD circuit is defined as the number of clock cycles elapsed from the time the input is presented until the output becomes available. It is equal to the maximum number of basic gates connected in any path from input to output.

Definition 2. The cost of a CCD circuit is a measure of the chip area occupied by the basic gates needed in the circuit. Every basic gate has been assigned a cost factor [7], as shown in Fig. 1.

TERNARY SYSTOLIC ARRAY PROCESSOR

The systolic architectural concept was developed by Kung and his colleagues [1,10]. Since then, various systolic array processors have been designed, such as, one-dimensional linear arrays, two dimensional square arrays and hexagonal arrays which are often applied to matrix computations. The hexagonal array architecture is used in the proposed ternary systolic array processor. The structure is the same as in [10].

The core of the systolic array processor is the inner-product step processor. A ternary CCD inner-product step processor was designed in [11]. This step processor can implement the desired computations:

$$R = A * B + D \tag{1}$$

However, the number of clock periods, where the computations of the components of the result R take, are different. In order to implement a desired step processor, as shown in Fig. 3, one must have

$$C_{out}^{k+1} = C_{in}^{k} + A_{in}^{k} * B_{in}^{k} \tag{2}$$

$$A_{out}^{k+1} = A_{in}^{k} \tag{3}$$

$$B_{out}^{k+1} = B_{in}^{k} \tag{4}$$

where A_{in}^{k}, B_{in}^{k}, and C_{in}^{k}, are inputs of the step processor at step k and A_{out}^{k+1}, B_{out}^{k+1}, and C_{out}^{k+1} are outputs of the step processor at step k+1. To implement Eq. (2)-(4), some delay blocks are put in the block diagram as shown in Fig. 4. Because the depth of the inner-product step processor in [11] is 52 clock periods, the step length should be 52 clock periods too. The a_i and b_i which are components of A and B are also delayed for 52 clock periods (Fig. 4).

The relative costs of the components and the step processor were investigated and the results are listed in Table 1.

The revised inner-product step processor can be applied to different configurations of systolic array processors. The hexagonal array architecture will be shown in detail in the next section.

The major drawback of CCD circuits is the low speed [7]. The step length of the proposed step processor is quite long. However, in comparison with the matrix multiplication implemented in a single CCD processor, the parallel and pipeline processing in the proposed systolic array processor speeds up the matrix computation significantly.

TABLE 1 RELATIVE COSTS OF SOME CCD CIRCUITS

circuits	cost
adder with carry	14
delay block of n clocks	2n
HMA	66
FMA	82
step processor of 4 digits	1748

FAULT-TOLERANCE DESIGN OF TERNARY SYSTOLIC ARRAY PROCESSORS

To achieve high reliability, various fault-tolerant techniques can be applied to the designed array processor. So-called algorithm-based fault tolerance, which can detect and correct errors caused by permanent or transient failures in VLSI hardware with a suprisingly low overhead, was recently proposed [12,13]. This fault tolerance approach is more suitable for single processor failure instead of gate-level single stuck-at faults. This approach is applied to the proposed ternary systolic array processor.

The principles of algorithm-based fault tolerance and its application to matrix operation were discussed in detail [13]. Here, some basic facts and its implementation in ternary CCD circuits are investigated.

1. Checksum matrices and ternary adder:

The column checksum matrix A_c of an n-by-m matrix A is an (n+1)-by-m matrix expressed by $A_c = \left[\begin{array}{c} A \\ \hline e^T A \end{array}\right]$ where e^T is a 1-by-n vector [11 . . . 1] and $e^T A$ are column checksums. The row checksum matrix A_r of an n-by-m matrix A is an n-by-(m+1) matrix expressed by $A_r = [A|Ae]$ where e is a m-by-1 vector $[\,11\,.\,.\,.\,1]^T$ and Ae are row checksums. The full checksum matrix A_f of an n-by-m matrix A is an (n+1)-by-(m+1) matrix expressed by $A_f = \left[\begin{array}{c|c} AB & AB\underline{e} \\ \hline e^T AB & e^T ABe \end{array}\right]$. The result of a column checksum matrix A_c multiplied by a row checksum matrix B_r is a full checksum matrix C_f:

$$A_c * B_r = \left[\begin{array}{c} A \\ \hline e^T A \end{array}\right] * \left[\begin{array}{c|c} B & Be \end{array}\right] = \left[\begin{array}{c|c} AB & ABe \\ \hline e^T AB & e^T ABe \end{array}\right] = C_f$$

where ABe are row checksums, $e^{T}AB$ are column checksums, and $e^{T}ABe$ are checksum of row checksums and column checksums.

In order to compute checksum and the difference between the sum and the checksum, a CCD modulo-3 adder is designed. A four digit ternary adder which consists of 4 cascaded, two-addendum full adders is shown in Fig. 5. The depth of the adder is 10, and the cost is 56. The checksum computation can be implemented by combining inner-product step processors and modulo-3 adders, as shown in Figure 7.

2. Error Detection, Correction, and Ternary Subtractor

Error detection, location and correction are carried out by using column checksum and row checksum. The approaches are as follows:

Error detection: Compare the sum of each row and each column with the corresponding row checksum and column checksum respectively. Any inconsistence in a row or a column determines an error.

Error location: Any error can cause the inconsistence both in a row and in a column. The location can be determined by the intersection of that row and that column.

Error correction: The erroneous element can be corrected by adding the difference of the sum and checksum (in the inconsistant row or column) to the erroneous element.

Based on the above discussion, the comparison and subtraction of a sum and a checksum should be carried out. Instead of constructing a ternary CCD subtractor, 3's complement addition which uses complement circuits and modulo-3 adder is applied. The 3's complement addition is similar to the 2's complement addition in the binary system. A CCD 2's complement circuit is designed and shown in Fig. 6. This circuit can implement the the following conversion: 0 → 2, 1 → 1, and 2 → 0. The depth of the complement circuit is 3 and the cost in 19. The comparison result can be generated by adding all the digits of the difference using an addition gate. If the output of the addition gate is zero, then no error occurs; any non-zero output indicates the inconsistence of the sum and checksum.

3. Fault Tolerant Systolic Array Processor.

The fault tolerant, ternary, systolic array processor proposed in this paper is similar to the processor proposed by Huang and Abraham [13]. The whole system configuration is shown in Fig. 7. The ternary modulo-3 adders and the complement blocks are used in the fault tolerant block F1, F2, F3, and F4. If the band width of matrices A and B are 4, the cost of F1 and F2 (not including the 4 step processor units) is around 600 and the cost of F3 and F4 is around 700. The hardware for fault tolerance used in this design has less cost than any other fault tolerance techniques.

CONCLUSION AND DISCUSSION

A fault tolerant, ternary, CCD systolic array processor is proposed in this paper. The computation time of matrix multiplication in this processor is much less than that in a single CCD processor. Algorithm-based fault tolerance is applied to this CCD array processor to ensure that the results of long computations are valid. In principle, the design approach used here can be applied to any radix, multivalued CCD array processors. The fault tolerance design for floating point operations in the CCD array processor needs to be further developed.

REFERENCES

[1] Brent, R.P. and Kung, H.T., "The Chip Complexity of Binary Arithmetic," Proc. 12th. Annual ACM Symp. Theory of Computing, Los Angles, CA, April 1980, pp. 190-200.

[2] Smith, K.C., "The Prospects for Multiple-Valued Logic: A Technololgy and Application View," IEEE Trans. Comp., Vol. C-30, No. 9, Sept. 1981, pp. 619-634.

[3] Kerkhoff, H.G. and Tervoert, M.L., "The Implementation of Multi-Valued Functions Using CCD's," Proc. 10th. ISMVL, 1980, pp. 6-15.

[4] Kerkhoff, H.G. and Tervoert, M.L., "Multiple-Valued Logic CCD's," IEEE Trans. Comp., Vol. C-30, Sept. 1981, pp. 644-652.

[5] Tirumula, P. and Butler, T.J., "On the Realization of Multi-Valued Logic Functions Using CCD PLA's," Proc. 14th. ISMVL, 1984, pp. 33-42.

[6] Tomabechi, N., Kameyama, M., and Higuchi, T., "Pulse Train Residue Arithmetic Circuit Using Multiple-Valued CCD's and Its Application to Digital Filters," Proc. 13th. ISMVL, 1983, pp. 146-151.

[7] Abd-el Bar, M.H., Zaky, S.G., and Vranesic, Z.G., "Synthesis of Multivalued Multithreshold Functions for CCD Implementation," IEEE Trans. Comp., Vol. C-35, No. 2, Feb. 1986, pp. 124-133.

[8] Kerkhoff, H.G. and Robroek, H.A.J., "The Logic Design of Multi-Valued Logic Functions Using CCD's," Proc. 12th. ISMVL, 1982, pp. 35-44.

[9] Hwang, K. and Briggs, F.A., Computer Architecture and Parallel Processing, McGraw-Hill, Inc., N.Y., 1984.

[10] Kung, H.T. and Leiserson, C.E., "Algorithms for VLSI Processor Arrays," in Introduction to VLSI Systems, C.A. Mead and L.A. Conway, Eds. Reading, MA: Addison-Wesley, 1980, pp. 271-292.

[11] Manzoul, M.A. and Han, J.-Y., "A Ternary CCD Inner Product Step Processor," submitted to Compcon, Spring 1987, San Francisco, CA.

[12] Banergee, P. and Abraham, J.A., "Bounds on Algorithm-Based Fault Tolerance in Multiple Processor Systems," IEEE Transactions on Computers, Vol. C-35, No. 4, April 1986, pp. 296-306.

[13] Huang, K.H. and Abraham, J.A., "Algorithm-Based Fault Tolerance for Matrix Operations," IEEE Transactions on Computers, Vol. C-33, June 1984, pp. 518-528.

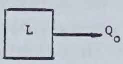
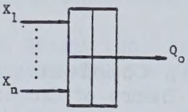
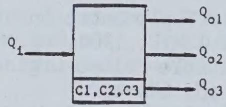
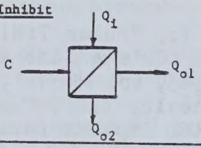
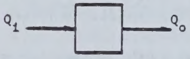
Symbols	Logic Function	Relative Cost
1. **Constant** 	$Q_o = L$ $L=1,2,\ldots$	1
2. **Addition** 	$Q_o = \sum_{i=1}^{n} X_i$	2n-2
3. **Fixed Overflow**  Q_i Q_{o1} Q_{o2} Q_{o3} C1,C2,C3	$Q_{o1} = \min(Q_i, C_1)$ $Q_{o2} = \min(Q_i - C_1, C_2)$ $Q_{o3} = \min(Q_i - C_1 - C_2, C_3)$	4
4. **Inhibit** 	$Q_{o1} = Q_i$ if $C \neq o$, o otherwise $Q_{o2} = Q_i$ if $C = o$, o otherwise	6 if $Q_i \leq 2$ 18 if $Q_i \geq 3$
5. **Delay** 	$Q_o(t) = Q_i(t-1)$	2

Figure 1. Basic CCD Gates

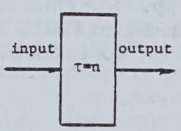

input
output
τ=n

Figure 2. Symbol of a Delay Gate of n Clock Periods

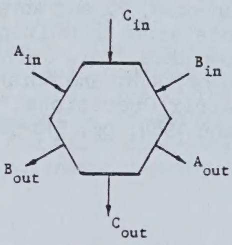

C_{in}
A_{in}
B_{in}
B_{out}
A_{out}
C_{out}

Figure 3. Step Processor in a Hexagonal Array Processor

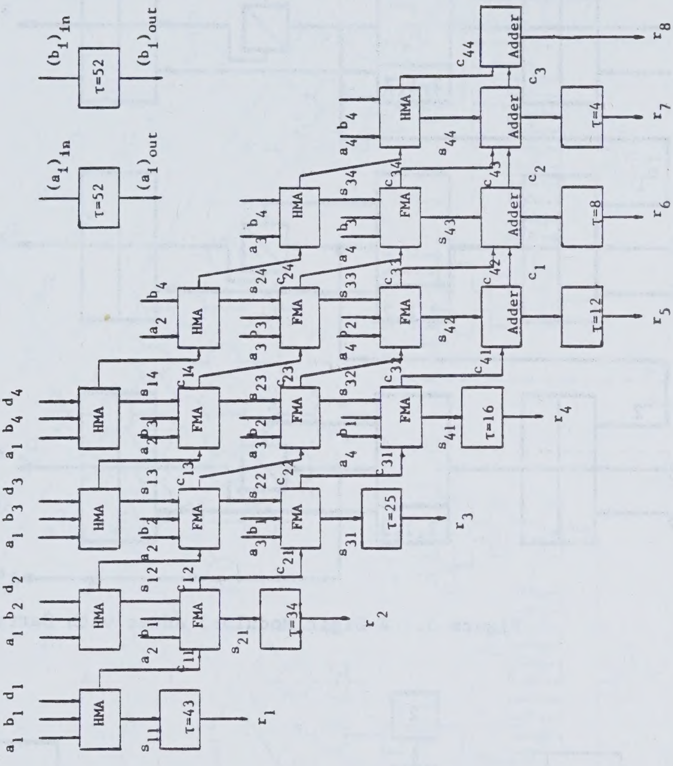

$(a_1)_{in}$
τ=52
$(a_1)_{out}$
$(b_1)_{in}$
τ=52
$(b_1)_{out}$
a_1 b_1 d_1
a_1 b_2 d_2
a_1 b_3 d_3
a_1 b_4 d_4
HMA
FMA
Adder
τ=43
τ=34
τ=25
τ=16
τ=12
τ=8
τ=4
s_{11}
s_{12}
s_{13}
s_{14}
s_{21}
s_{22}
s_{23}
s_{24}
s_{31}
s_{32}
s_{33}
s_{34}
s_{41}
s_{42}
s_{43}
s_{44}
c_{11}
c_{12}
c_{13}
c_{14}
c_{21}
c_{22}
c_{23}
c_{24}
c_{31}
c_{32}
c_{33}
c_{34}
c_{41}
c_{42}
c_{43}
c_{44}
c_1
c_2
c_3
a_2 b_1
a_2 b_2
a_2 b_3
a_2 b_4
a_3 b_1
a_3 b_2
a_3 b_3
a_3 b_4
a_4 b_1
a_4 b_2
a_4 b_3
a_4 b_4
r_1
r_2
r_3
r_4
r_5
r_6
r_7
r_8

Figure 4 Block Diagram of Inner Product Step Processor

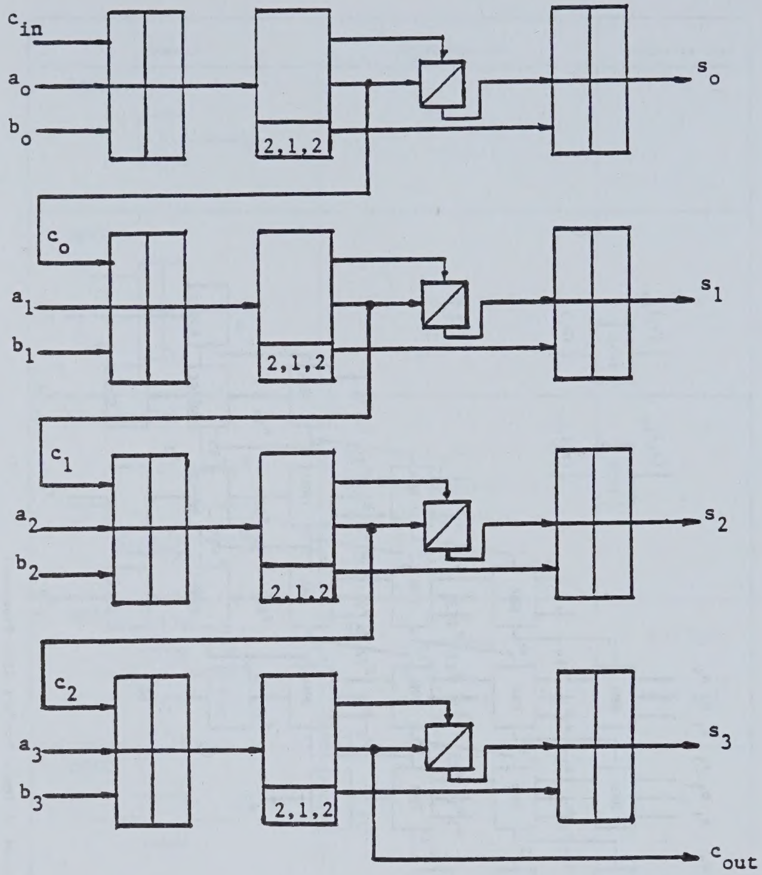

c_in
a_o
b_o
2,1,2
s_o
c_o
a_1
b_1
2,1,2
s_1
c_1
a_2
b_2
2,1,2
s_2
c_2
a_3
b_3
2,1,2
s_3
c_out

Figure 5. 4 Digit Modulo-3 Adder with Carry

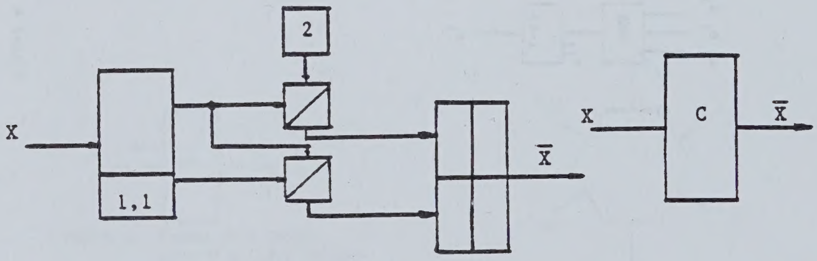


Figure 6. 2's Complement Circuit and Block for One Digit, Modulo-3 Number

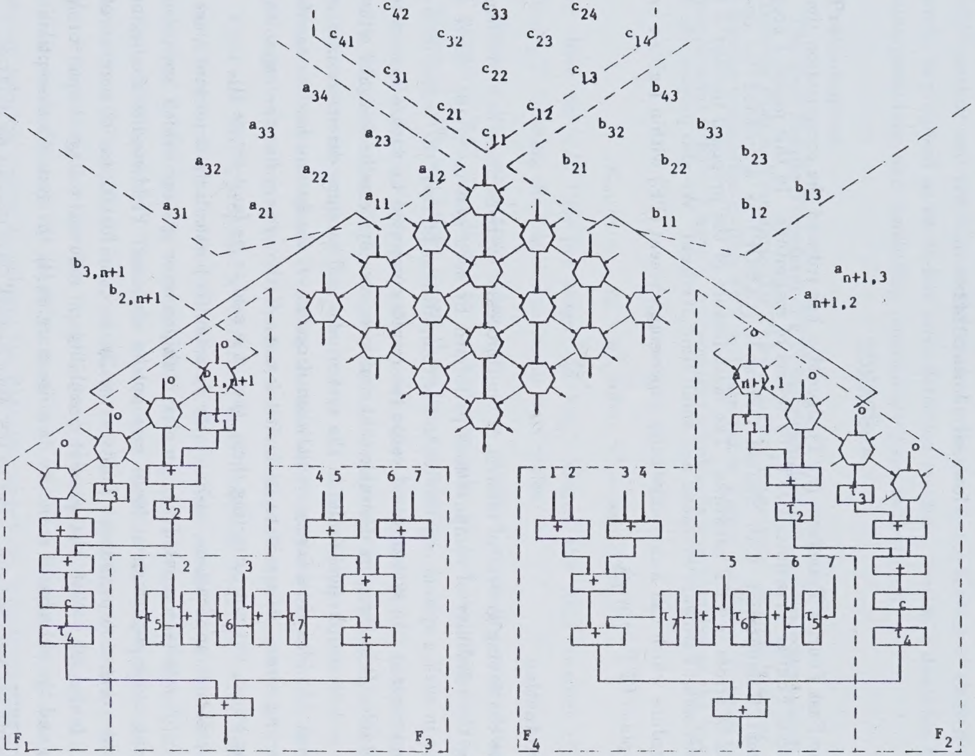

c42
c33
c24
c41
c32
c23
c14
c31
c22
c13
a34
c21
c12
b43
a33
a23
c11
b32
b33
a32
a22
a12
b21
b22
b23
a31
a21
a11
b11
b12
b13
b3,n+1
b2,n+1
an+1,3
an+1,2
b1,n+1
an+1,1
o
τ1
τ2
τ3
τ4
τ5
τ6
τ7
+
c
1 2 3 4 5 6 7
F1
F2
F3
F4

Figure 7. Architecture of the Fault Tolerant Systolic Array Processor

Concurrent Error Detection in FFT Networks†

Jung-Hwan Kim

Department of Electrical and Computer Engineering
University of Iowa
Iowa city, Iowa 52242

ABSTARCT

The Fast Fourier Transform (FFT) algorithm has reduced the computation time of the Discrete Fourier Transform (DFT) by orders of magnitude. In this paper, I propose Concurrent Computation with Shifted OPerands (CCSOP) as a method for on-line detection of errors in FFT networks. The main feature of the proposed method is that it requires small hardware overhead and small time overhead. We also propose the testing procedure which can locate any faulty processing element (PE) within logN+4 cycles for a N-point FFT network.

1. Introduction

The continuing growth of interest in multiprocessor systems poses new problems in ensuring the reliability of computations performed by such systems [7-10]. Since each processor in such a system contributes to the computation process, the failure of even a single processor in the system could cause the results computed by such a system to be unacceptable. It is therefore desirable that multiprocessor systems be designed either to be able to tolerate physical faults in the system and still produce correct results, or at least be able to identify a faulty result when it occurs. If the system has the capability of correcting errors, it is said to be fault-tolerant. If the system has the capability of detecting errors, but not correcting them, then it is said to be fault-secure [1].

In real-time applications, on-line error detection is particularly important since the reliability of results computed by complex multiprocessor systems which are prone to permanent and non-permanent faults may not be adaquate. Field studies are beginning to indicate that non-permanent faults typically account for 90 to 98 percent of all detected faults [2]. With the present possibility of reduced voltage levels for VLSI modules and the subsequent reduction in noise margins [4], the system's susceptibility to

---

† The research has been supported in part by U.S. Army Research Office Contract No. DAAG29-86-K-0066.

transient faults also increases. Therefore, techniques are needed to detect faults concurrently with normal operation, especially in real-time applications.

In this paper, concurrent error detection schemes in homogeneous multiprocessor systems are given. Specifically I consider Fast Fourier Transform (FFT) networks. The remainder of this paper is organized as follows. Preliminary concepts are developed in Section 2. Section 3 describes the proposed on-line error detection method, called CCSOP method and the fault-model used. In Section 4, application of CCSOP to FFT networks is proposed as an on-line error detection method. Section 5 describes the fault location procedure, and concluding remarks are given in Section 6.

2. Preliminaries

The Fast Fourier Transform (FFT) algorithm has reduced the computation time of the Discrete Fourier Transform (DFT) by orders of magnitude. A complete flow diagram of a 8-point decimation-in-time FFT is illustrated in Figure 1. Figure 2 shows each processing element (PE) of Figure 1, representing a butterfly computation of the form,

$$x_{out} = x_{in} + y_{in} * w^{k}$$

$$y_{out} = x_{in} - y_{in} * w^{k}, \text{ where } w^{k} = \exp(-i2\pi k/N)$$

Since there are n/2 PEs per stage and $\log_2 N$ stages, the total number of PEs is $(N*\log N)/2$, finishing the transformation in logN cycles.

Recently two main approaches have been proposed as on-line error detection in FFT networks [5,6]. One is due to Jou and Abraham [6], and the other is due to Choi and Malek [5]. We will briefly review these two approaches.

Jou and Abraham proposed a Concurrent Error Detection (CED) method for a FFT network which encodes the input sequence and decode the output sequence to give each output a nontrivial weighted contribution to the checksum [6]. CED scheme requires hardware and time overhead involving the checksum computation, time for summation of the output, encoding of input sequence, and decoding of output sequences.

Choi and Malek proposed a concurrent error detection technique, called Recomputing by Alternate Path, for an FFT network [5]. In this scheme, a duplicated computation delayed by one cycle is executed with rearranged input sequence through alternate path in a FFT network. Therefore this scheme requires the rearrangement of input sequence for the duplicated computation as well as a capability to switch paths at the output. Depending on how the input and output sequence rearragements are achived the area overhead could be small to high, with the time overhead depending on the hardware solution chosen to achieve the data rearrangement, bypass lines and switches at the inputs and outputs in every PE.

In this paper, I propose the application of Concurrent Computation with Shifted OPerands (CCSOP) method to FFT networks as an on-line error detection method. The main feature of the proposed method is that it requires small hardware overhead and small time overhead.

3. Description of CCSOP and Fault Model

In this section I briefly explain the CCSOP method being proposed and the fault model used. CCSOP uses the principle of time redundancy of Figure 3, where the coding function C is the shift left operation and the decoding function C^{-1} is the shift right operation. Thus, in the first computation step, at time t=1, X is passed through the shifter unshifted and then input to unit F_1. At t=n, the result of unit F_n is passed through the shifter unshifted and stored in a register. During a second and concurrently occurring computation step, at t=2, X is shifted left by k bits and then input to unit F_1. At t=n+1, the result of unit F_n is shifted right by k bits and compared to the result of unit F_n in the first step. A mismatch indicates an error in either computation step. CCSOP method can be easily understood in Figure 5, which shows the application of CCSOP to the FFT networks of Figure 1. The bold-faced letters in Figure 5 indicate left-shifted operands.

CCSOP method being proposed can be seen to be similar to the RESO method proposed earlier in [11,12]. The main difference between RESO and CCSOP is the *concurrent* nature of computations in CCSOP, while in RESO a computation is completed and then a recomputation with shifted operands is performed, thus requiring 100% time overhead and some hardware overhead to hold all the results of the first computation and compare them with the results of the second computation. In CCSOP each duplicate computation requires only one more clock cycle regardless of the number of clock cycles needed in the first computations, thus depending on applications time overhead can be negligible. For example, in a N-point FFT network time overhead is $O(\frac{1}{\log N})$.

In this paper, only operational faults are considered. The manufacturing defects are assumed to have already been tested through off-line testing. We consider only the single fault case. That is either only one adder-cell which is a constituent of an adder and a multiplier or one bit data line is faulty during the operation. The input faults to an adder-cell are regarded as a fault in the adder-cell since the fault behaviour can be included into that of the adder-cell. Similarly, a fault in the register stage is regarded as a data line fault. The fault model I assume here is a functional fault model of one adder-cell or a data line. In [5,7,10] the fault model used was PE-level functional fault, i.e. a PE may produce any erroneous result, and in [6] the fault model was that of functional faults occurring in an element such as a single multiplier or adder or one set of input or output lines. The proposed fault model, therefore, is limited compared to the earilier models, but leads to low overhead solutions. Additionally it is believed that the single element fault model used here, would provide coverage for all single PE faults with very high probability.

The comparator may be implemented as a self-checking circuit based on dual rail codes [14], in which case this unit can be assumed to be prone to failures and such failures will also be detected on-line. The data w^k is assumed to be hard-cored. Since one faulty bit of the data w^k affects several adder-cells our fault model can not allow the fault in the data w^k. In this paper, I assume that a PE consists of a ripple-carry adder or a carry-lookahead adder (CLA) and a multiplier with the structure using ripple-carry adders or a Wallace tree [3]. In CLA an adder-cell i is defined to consist of units f_i and g_i of Figure 4.

4. Application of CCSOP to FFT Networks

In this section, the application of CCSOP to fixed-point as well as floating-point operations of FFT networks as an on-line error detection scheme is discussed.

4.1. Application to Fixed-Point Operations

Fixed-point operations of FFT networks are considered in this subsection. Figure 5 shows the application of CCSOP to the FFT networks of Figure 1. CCSOP-k is defined to be the name of error detection scheme achieved by recomputing with the input sequence x_is shifted by k bits. For example, in Figure 5 only input sequence x_is, not w^ks are shifted during the recomputation step.

The proposed on-line error detection scheme is as follows. During the first computation step, the computation is done with unshifted operands and the result through the last stage is stored in a register. During a second and concurrently occurring recomputation step , the computation is done with left-shifted operands x_i, and the complete results x_is through the last stage are stored in a register. The contents of both registers are compared by choosing 1 through m bits of the first computation and 3 through m+2 bits of that of the recomputation for CCSOP-2 scheme of m bits operations. A mismatch indicates an error in either computation step. Here the operands x_is for the recomputing step are left-shifted before computing.

Lemma 1 : CCSOP-2 detects all errors produced by a single fault in fixed-point operations of a FFT network.

Proof : Any single fault in a FFT network can be classified to belong to the following two cases since the data w^k is assumed to be fault-free.

Case 1: One adder-cell is faulty.

Let one adder-cell whose sum bit has weight 2^i and the carry-out 2^{i+1} be faulty in stage j. The sum and carry-out of the faulty adder-cell can have any logical value at any time. During the first computation step, the sum and carry-out of the faulty adder-cell in stage j have weights 2^i and 2^{i+1}, respectively. Therefore the partial result in stage j is off the correct one by one of $\{0, \pm 2^i, \pm 2^{i+1}, \pm 3*2^i\}$. This error is multiplied by w^k, $0 \leq k < N$, factor in each stage from stage (j+1) to the last stage logN-1. Each final result affected by the faulty adder-cell is off the correct one by one of $\{0, \pm w^k*2^i, \pm w^k*2^{i+1}, \pm w^k*3*2^i\}$, where $0 \leq k < N$, since $w^N = 1$. During the recomputation step, the sum and carry-out of the faulty adder-cell have weights 2^{i-2} and 2^{i-1}, respectively due to the shifted operands. Therefore the partial result of the recomputation step at stage j is off the correct one by one of $\{0, \pm 2^{i-2}, \pm 2^{i-1}, \pm 3*2^{i-2}\}$. This error is multiplied by w^k, $0 \leq k < N$, factor in each stage from stage (j+1) to the last stage logN-1. Each final result affected by the faulty adder-cell is off the correct one by one of $\{0, \pm w^k*2^{i-2}, \pm w^k*2^{i-1}, \pm w^k*3*2^{i-2}\}$, where $0 \leq k < N$, since $w^N = 1$. The two computations will have identical results only when there are no errors. Therefore, any error in either step will cause a mismatch of the results of the two compuation steps, and the error is detected.

Case 2: One bit line of X-data is faulty.

Let one bit which has weight 2^i be faulty. During the first and second computation steps, the faulty line has weights 2^i and 2^{i-2}, respectively. During the first and second computation steps, the result is off the correct result by one of $\{0, \pm w^k*2^i\}$ and by one of $\{0, \pm w^k*2^{i-2}\}$, where $1 \leq k \leq N-1$, respectively. The two computations will have identical results only when there are no errors. Therefore, the error is detected. <Q.E.D>

The hardware overhead of CCSOP-2 for a FFT network is quite small. For example, for 32-bit operations a PE of CCSOP-2 needs six 34-bit adders and four modified 34-bit multipliers (only one operand is shifted) since a butterfly computation involves four real multiplications and six real additions. In addition, the CCSOP-2 array needs N 32-bit comparators. There are 32x32 adder cells for a 32-bit multiplier and a modified 34-bit multiplier has as many adder-cells as a 33-bit multiplier. Assumming that each bit of the comparator is of the same complexity as that of an adder-cell, hardware overhead is

$$\frac{(33^2-32^2)*4 + (34-32)*6 + \frac{2*32}{n}}{(\ 32^2*4 + 32*6\ )}$$

where n=logN. For 32-point FFT (n=5), hardware overhead is about 6.6%. The time overhead in the N-point FFT network is 1/logN.

4.2. Application to Floating-Point Operations

In this subsection, the application of CCSOP to floating-point operations is proposed. Figure 6(a) and 6(b) show an adder and a multiplier expressed by floating-point operations. For floating-point operations, one can apply the already established CCSOP techniques for fixed-point operations.

The proposed on-line error detection scheme is as follows. During the first computation step, the computation is done with unshifted exponents and unshifted mantissas, and the final result through the last stage is stored in a register unshifted. During the first computation step, mantissas of all input sequence have two leading 0 bits, and the 0-digit check function unit allows the mantissas to have two leading 0 bits. During the recomputation step, the computation is done with exponents and mantissas of all input sequence and exponent of all W factor (to align with input sequence) shifted left by 2 bits. Shift-left (SL2) and shift-right (SR2) operations in Figure 6(a) and 6(b) are also activated during recomputation step to adjust the difference between the exponent and the mantissa due to shifted operands. The final recomputation results, exponents and mantissas, through the last stage are each shifted right by 2 bits and stored in registers. The contents of both registers holding mantissas and the contents of both registers holding exponents are compared by choosing bits in a similar way described in subsection 4.1. A mismatch in either comparison indicates an error in either computation step.

Lemma 2 : CCSOP-2 detects all errors in a FFT network expressed by floating-point operations, when failures are confined to one adder-cell which is a constituent of an adder (subtractor) and a multiplier.

Proof: Case 1 : One adder-cell of a subtractor unit in Figure 6(a) is faulty.

Let one adder-cell whose sum bit has weight 2^i and the carry-out 2^{i+1} be faulty. The sum and carry-out of the faulty adder-cell can have any logical value at any time. During the first computation step, the sum and carry-out of the faulty adder-cell have weights 2^i and 2^{i+1}, respectively. Therefore the exponent result of a subtractor unit is off the correct one by d_1, which is one of $\{0, \pm 2^i, \pm 2^{i+1}, \pm 3*2^i\}$. Since the faulty exponent computation in a subtractor unit produces the error in the mantissa part, the faulty PE's mantissa result $(M_a 2^{E_a-E_t+d_1} - M_t)$ is off the correct mantissa $(M_a 2^{E_a-E_t} - M_t)$ by $M_a 2^{d_1}$, which is also the error of the final mantissa result, assuming $E_a \leq E_t$. During the recomputation step, the exponent result of the substractor unit and SL2 unit (active during the recomputation step) is off the correct one by d_2, which is one of $\{0, \pm 2^{i-2}, \pm 2^{i-1}, \pm 3*2^{i-2}\}$. Due to the faulty exponent computaion, the final mantissa result through the last PE, PE[n] is off the correct one by $M_a 2^{d_2}$. The two computations will have identical results only when there are no errors. Therefore, any error in exponent or mantissa part during either step will cause a mismatch of the results of the two compuation steps, and the error in an adder-cell is detected.

The other cases for one faulty adder-cell in other adders and multipliers can be proved similarly to the case 1 of Lemma 1.

5. Fault Location

In this section, I discuss how to locate a faulty PE after an error is detected. The modified PE which has a bypass line is shown in Figure 7. A fault in a PE at stage j propagates an error to every node of the subtree rooted by that PE. Since the error is detected and the test input sequence which sensitizes the fault to produce the error is known, the PE which has a permanent fault can be located by some additional computations.

Definition 1 : The jth PE at stage i in the FFT network is called PE[i,j].

DEfinition 2 : If PE[i-1,j] is directly connected to PE[i,k], then PE[i-1,j] is called the *predecessor* of PE[i,k], and PE[i,k] is called the *successor* of PE[i,j-1].

A fault in a multiplier of a PE at stage j has the same syndrome as that in an adder of its predecessors at stage j-1. Therefore, to locate a faulty PE, the ambiguity between stage (j-1) and stage j must be solved.

The proposed fault location scheme is as follows. From k=(the number of mismatching output), $1 \leq k \leq N$, I know that a multiplier in stage s+1 or an adder in stage s is faulty, where s = n–logk–1. For example, if k=4 for a 8-point FFT, a multiplier in the second stage or an adder in the first stage is faulty since s=0. In order to test whether the multiplier at stage s+1 is faulty or not, at t=T+1 and t=T+2, do FFT computation with unshifted and shifted input sequence, respectively, by bypassing the stage s+1. Then to test the adder at a stage s, at t=T+3 and t=T+4, do FFT computation with unshifted and shifted input sequence, respectively, by bypassing the stage s. At t=logN+2, both results with the bypassed stage s+1 are compared. If the comparison match, then all multipliers (thus PEs) at stage s+1 is fault-free. Otherwise, the faulty multiplier at stage s+1 can be located by backtracking the mismatched output. At t=logN+4, both results with the bypassed stage s are compared. If the comparison match, then all adders (thus PEs) at stage s is fault-free. Otherwise, the faulty adder at stage s can be located by backtracking the mismatched output. Therefore, any faulty PE can be located within logN+4 cycles. The fault location method is formally described as PROCEDURE FAULT-LOCATE. From the above discussion, the following Theorem can be formally stated.

Theorem 1 : PROCEDURE FAULT-LOCATE can locate any faulty PE within logN+4 cycles in a N-point FFT network.

```
PROCEDURE FAULT-LOCATE
          N: number of inputs ( n=logN)
          S: the possible faulty stage
          K: the number of mismatching outputs
          X: unshifted input sequence
          X': shifted input sequence
          FFT(T,X,BP(s)): At time T, do FFT computation
          with input X by bypassing stage s.

          BEGIN
               S = n - logK - 1
               $FFT_1$ (T+1, X, BP(s+1))
               $FFT_2$ (T+2, X',BP(s+1))
               $FFT_3$ (T+3, X, BP(s))
               $FFT_4$ (T+4, X', BP(s))
               Compare (T+n+2, $FFT_1$, $FFT_2$)
               Compare (T+n+4, $FFT_3$, $FFT_4$)
          END
```

6. Concluding Remarks

In this paper, I have proposed on-line error detection scheme in FFT networks. The proposed method, Concurrent Computataion with Shifted OPerands (CCSOP), is suitable for FFT networks since it require small hardware overhead and small time overhead. Although FFT networks have been considered in this paper, the method can be applied to other homogeneous multiprocessing system such as systolic arrays. Also, I presented a fault location scheme which can locate any faulty PE within logN+4 cycles in a N-point FFT network.

REFERENCES

[1] P. Banerjee and J.A. Abraham, "Fault-Secure Algorithm for Multiple Processor Systems," *Porc. 11th Symp. on Computer Architecture,* June 1984, pp.279-287.

[2] S.R. McConnel, D.P.Siewiorek and M.M. Tsao, "The measurement and Analysis of Transient Errors in Digital Computer Systems," *Proc. 9th Intl. Symp. on Fault Tolerant Computing,* June 1979, pp.67-70.

[3] J.-L. Baer, *Computer Systems Architecture,* Computer Sience Press, 1980.

[4] D.F.Barbe, "VHSIC Systems and Technology," *IEEE Computer,* Feb. 1981, pp.13-22.

[5] Y.H.Choi and M.Malek, "A Fault Tolerant FFT networks," *Proc. 15th Intl. Symp. on Fault Tolerant Computing,* June 1985, PP. 266-271

[6] J.Y.Jou and J.A.Abraham, "Fault Tolerant FFT networks," *Proc. 15th Intl. Symp. on Fault Tolerant Computing,* June 1985 PP.338-343

[7] Y.H.Choi, S.H.Han and M.Malek,"Fault-Diagnosis of Reconfigurable Systolic Arrays," *1984 Intl. Conf. on Computer Design,* Oct. 1984, pp.451-455.

[8] D.Fussel and P.Varman, "Fault tolerant wafer scale architechure for VLSI," *Proc. 9th Symp. on Computer Architecture,* April 1982, pp.190-198.

[9] J.H.Kim and S.M.Reddy, "A fault-Tolerant Systolic Array Design using TMR Method," *1985 Intl. Conf. on Computer Design,* Oct. 1985, pp.769-773.

[10] K.H.Huang and J.A.Abraham, "Algorithm-based Fault-Tolerance for Matrix Operations," *IEEE Trans. on Computer,* Vol C-33, no.6 pp.516-528

[11] J.H.Patel and L.Y.Fung, "Concurrent Error Detection in ALUs by Recomputing with Shifted Operands," *IEEE Trans. on Computers,* Vol C-31, No 7, July 1982, PP.589-595

[12] J.H.Patel and L.Y.Fung, "Concurrent Error Detection in Multiply and Divide Arrays," *IEEE Trans. on Computers,* Vol C-32, No.4, April 1983, PP. 417-422

[13] I.Koren, "Comments on 'The Diogenes Approach to Testable Fault-Tolerant Arrays of Processors'," *IEEE Trans. on Computers,* Vol C-35, No.1, Jan. 1986, P.94

[14] J.Wakerly, "Error Detecting Codes, Self Checking Circuits and Applications," *North Holland,* 1978.

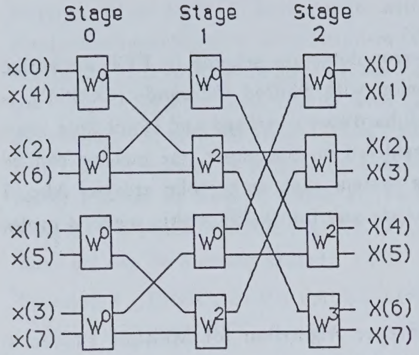

Stage 0
Stage 1
Stage 2
x(0)
x(4)
x(2)
x(6)
x(1)
x(5)
x(3)
x(7)
W^0
W^0
W^0
W^0
W^0
W^2
W^0
W^2
W^0
W^1
W^2
W^3
X(0)
X(1)
X(2)
X(3)
X(4)
X(5)
X(6)
X(7)

Figure 1. A 8-point FFT network

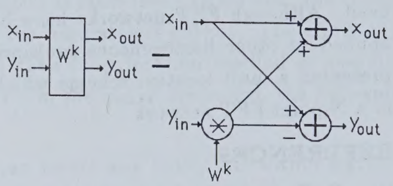

x_{in}
y_{in}
W^k
x_{out}
y_{out}
=
x_{in}
y_{in}
W^k
+
+
+
−
x_{out}
y_{out}

Figure 2. The function of a PE.

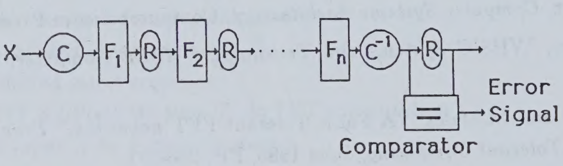

X
C
F_1
R
F_2
R
F_n
C^{-1}
R
Error Signal
Comparator

Figure 3. Description of CCSOP method.

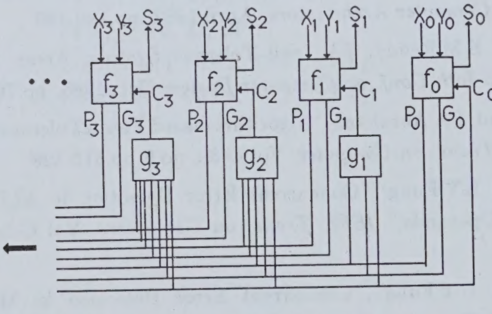

X_3 Y_3 S_3
X_2 Y_2 S_2
X_1 Y_1 S_1
X_0 Y_0 S_0
f_3
f_2
f_1
f_0
C_3
C_2
C_1
C_0
P_3 G_3
P_2 G_2
P_1 G_1
P_0 G_0
g_3
g_2
g_1

Figure 4. Typical implementation of a CLA.

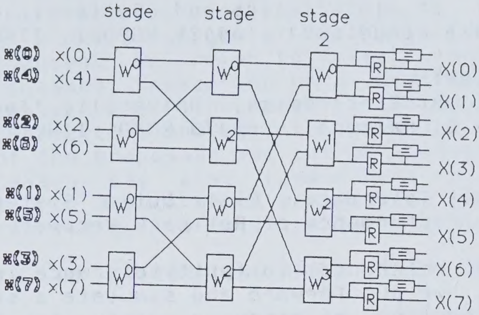

stage 0
stage 1
stage 2
x(0)
x(4)
x(2)
x(6)
x(1)
x(5)
x(3)
x(7)
x(0)
x(4)
x(2)
x(6)
x(1)
x(5)
x(3)
x(7)
W0
W0
W0
W0
W0
W2
W0
W2
W0
W1
W2
W3
R
=
X(0)
X(1)
X(2)
X(3)
X(4)
X(5)
X(6)
X(7)

Fig. 5. The CCSOP method in the FFT netwok

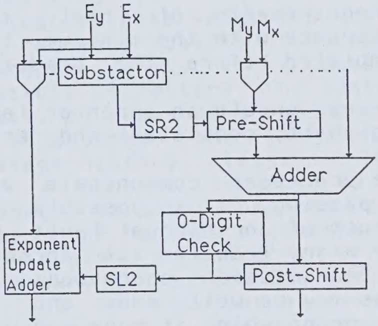

Ey
Ex
My
Mx
Substactor
SR2
Pre-Shift
Adder
0-Digit Check
Exponent Update Adder
SL2
Post-Shift

Figure 6(a). A floating-point adder

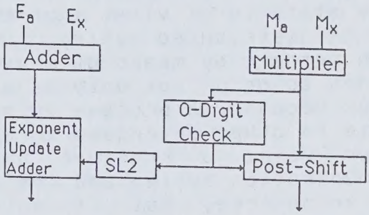

Ea
Ex
Ma
Mx
Adder
Multiplier
0-Digit Check
Exponent Update Adder
SL2
Post-Shift

Figure 6(b). A floating-point multiplier

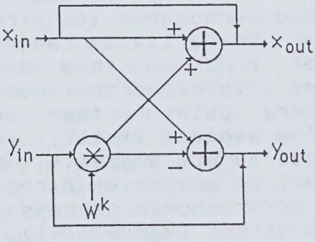

xin
xout
yin
yout
Wk
+
+
+
−

Figure 7. The modified function of a PE

AN APPROACH TO AN OPTIMAL STRATEGY OF RECOVERY POINT INSERTION IN DISTRIBUTED FAULT TOLERANT COMPUTING SYSTEMS

GIOVANNI CANTONE
Department of Informatics and Systems, University of Naples - via Claudio 21 - 80125 NAPOLI, ITALY

BRUNO CICIANI
Department of Electronics, University "Tor Vergata" of Rome - via O. Raimondi - 00173 ROMA, ITALY

keywords: Fault Tolerance, Distributed Systems, Backward Error Recovery, Performance of Rollback Recovery Mechanism.

Abstract: with reference to fault-tolerance in distributed process system, we put forward and simulate a strategy which allows the insertion of recovery points to be calibrated according to the process communication rate and process fault rates.

1. INTRODUCTION

One of the most important requirements of real time distributed systems is fault tolerance with the minimum time overhead so that tasks are completed before the deadline, even when faults are present.

In a system of interacting processes, should an error or fault be detected the recovery of a globally consistent and error free state is of vital importance .

In a distributed system, where processes communicate with each other by means of message passing, it is possible for errors to occur not only as a result of an internal fault but also because a process in error might transfer its erroneus state to other processes through transmission operations. In other words, a receiver process may be well made and the communication system and the machine on which it runs may well be error-free, but it could still be the object of error caused by the information it receives from one of its senders. On the other hand, since a process subject to such a communication error could, in turn, be the sender of other processes, the point is soon reached where an error in one process can rapidly spread throughout its direct and indirect receivers; in particular, the activity cancellations required after a propagation of error in this manner, can spread cyclically back to the original error source process and, thus, shift the recovery point further and further back towards the root of the process itself; in this way the erroneus state can spread to all segments, both active and terminated, which are part of processes directly or indirectly in partnership with the error source process and, thus, create a total rollback (Domino effect [Randell75]).

Many proposals have, however, been put forward to resolve the problem of constructing Domino effect free systems. Randell [Randell74] proposed the Conversation structure; this structure, however, introduces an extra-synchronization among the cooperating processes, which reduces the efficiency of the processes' execution. Russell [Russell80] examined the way to avoid the Domino effect in a processing environment where processes communicate with each other in an asynchronous manner only, without introducing extra-synchronizations. And there have been many other contributions to the solution of this problem; in [Lee84] a fault tolerant multiprocessor has been designed and evaluated using hardware recovery blocks and it has been shown that this proposal is efficient when the communication rate between processes is small.

In this paper a new strategy for the insertion of recovery points is proposed. We refer to tasks formed of a set of concurrent processes, executing on a distributed computing system. Processes are structured in recovery blocks as proposed in [Randell74] and used in [Lee84]. Our strategy is an extension of the proposals set out in [Russell80], in that firstly, processes may also communicate in a synchronous manner and not only asynchronously; secondly, a mark insertion operation is carried out to optimize the system throughput.

2. AN APPROACH TO CALIBRATING THE INSERTION OF RECOVERY POINTS

The two solutions put forward in [Russell80] for the elimination of the domino effect are denoted by MR and MRS, respectively, and concern concurrent processes which communicate by using Send and Receive asynchronous primitives.

Basically these solutions are based on the dynamic insertion of recovery points (System Recovery Points, SRP) which guarantee that, if a fault or error is detected, it is possible to restore the system to a consistent and error free state, by means of a domino effect free restoration operation. It is assumed that every process is equipped with a received message history, classified by senders; this avoids the need for a process subject to rollback having to re-acquire messages straight from its partner processes; indeed, in that case, the partner processes which were still "healthy" would be involved in the error process rollback because it would be necessary to re-send messages to the error process.

In the MR solution a state recovery operation (Mark) is developed on every Receive primitive. If an error is detected in a process, all the processes which have received messages straight from the process in error will have the received information and the activities dependent on it annulled; the annullment of activities as described above is, of course, to be extended to all segments importing error containing messages until that set of segments in error is empty. [Russell80] proves that MR (Mark each Receive) insertion strategy, which is described by the regular expression (Mark,Receive)*, is domino effect-free.

[Russell80] continues in its reference to asynchronous communications and proves that a system whose communications are protected by a MRS strategy, which is described by the regular expression (Mark, Receive*Send*)*, is domino effect free; this strategy does not pre-mark every receive with an SRP but rather pre-marks all the dynamic sequences formed of the following two types of communication blocks: firstly an r-block which contains at least one receive primitive and no send primitive; and secondly, an s-block which contains no receive primitives, but is equipped with send primitives (if the last communication of a segment is not a send the s-block is not necessary). It is clear that, when using MRS strategy, should an error in receive occur, it will be necessary to annul at least one r-block rather than annul just one Receive, as in the MR strategy (note that thanks to the Received Message Hystory, the annullment of the message received does not necessarily imply the re-sending of the message).

Figure 1 summarizes Russell's second marking strategy by means of a Mealy-like Flow State Machine (FSM).

To illustrate our proposal and so as to be congruent with Russell's proposals, we will, for the time being, only refer to the asynchronous environment, while the synchronous environment will be dealt with in the next section.

Let's consider the subseguent expression (1):

$$((M\ R[\leq D])^*S^*)^* \qquad (1)$$

where, referring to a process, $R[\leq D]$ denotes a dynamic sequence containing zero Send and from one to D Receive primitives and the "*" character denotes the traditional star (or repetitive) operation of a regular expression. Let's call the marking strategy defined by expression (1), the MRD strategy.

If we compare expression (1) with the $((MR)^*S^*)^*$ and the $(MR^*S^*)^*$ regular expressions proposed by Russell for marking processes, we can observe that

$$((M\ R[\leq D])^*S^*)^* \Longleftrightarrow ((MR)^*S^*)^* \quad \text{for } D = 1, \qquad (2)$$

and that

$$((M\ R[\leq D])^*S^*)^* \Longleftrightarrow (MR^*S^*)^* \quad \text{for } D \to \infty, \qquad (3)$$

and consequently that (1) is a more general expression than the other ones.

Furthermore, we observe that it can be easily proven that the MRD strategy is domino effect free; such a proof would be developed in an analogous way to that which was proven by Russell for the strategies described by expressions (2) and (3).

Fig.2 shows a Mealy-like FSM concerned with the mark insertion strategy described by expression (1).

Finally, expression (1) may be used to optimize the system throughput. It is effective but also useful because, by using expression (1) to save states, mark insertions may be calibrated between No.(2) and No.(3) extreme strategies by means of an oculated computation of the D parameter. On the other hand, expression (2) minimizes the time lost due to rollback but maximizes the overhead for saving the processing state, and, from this point of view expression (3) is exactely reciprocal to expression (2).

3. BACKWARD ERROR RECOVERY IN A SYNCHRONOUS / ASYNCHRONOUS ENVIRONMENT.

Our aim in this section is to extend the MRD strategy, as described in the previous section, to processes mutually communicating through synchronous message passing. To this end we will show, in due course, that such an extension is domino effect free.

Having already noted that MRD strategy is domino effect free, we can observe the following.

Given an L concurrent programming language in local environment, let C be the set of its communication primitives. If for each element of C there exists an equivalent primitive or program segment written in an LB language which contains only Send and/or Receive as communication primitives, then a system of processes written in L can be marked in such a way as to be domino effect free.

Indeed, under the above hypotheses it will be sufficient to mark the communication primitives of L, in every process, in accordance with criteria which are not less restrictive than those prescribed by the MRD strategy for the communication primitives present in the equivalent LB segment.

Let us consider the typical primitives in synchronous message passing. We denote a synchronous transmission operation as Call, and the corresponding reception operation as Accept. Moreover, we use the prefix "Trv" for a Tight Rendez-Vous synchronous communication (after message transmission, the resumption of processing of the sender is conditioned by the execution of the corresponding reception primitive of the receiver), and the prefix "Erv" for an Extended Rendez-Vous synchronous communication (the sender evolution is also conditioned by the processing and sending of output parameters from the receiver). Finally, we can say that a communication is Symmetric (Sym) if sender and receiver are exclusive owners of a communication channel, and Asymmetric (Asym) if a receiver draws messages from a channel which several senders can access; in the latter case, if the sender set is defined the communication is Tightly Asymmetric, elsewhere it is Loosely Asymmetric.

The following equivalences hold true for such definitions (we refer to the process sender with q and the receiver with p):

```
Sym_Trv_Call(p,mq)  primitive;
   begin send(p,mq); receive(p,[]) end;
Sym_Trv_Accept(q,mp) primitive;
   begin receive(q,mp); send(q,[]) end;
Sym_Erv_Call(p,m) primitive;
   begin send(p,mq.out); receive(p,[mq.in]) end;
Sym_Erv_Accept(q,mp) primitive;
   begin receive(q,mp.in); <local  activity>;
         send[mp.out]) end;
Asym_Trv_Call(p,mq) primitive;
   begin  send(p,mq);  receive(p,[]) end;
Asym_Trv_Accept(mp) primitive;
    begin receive(mp); acquire(transmitter);
          send(transmitter,[]) end;
Asym_Erv_Call(p,mq) primitive;
    begin send(p,mq.out); receive (p,[mq.in]) end;
Asym_Erv_Accept(mp) primitive;
    begin receive(mp.in); acquire(transmitter);
          <local_activity>;send(transmitter,[mp.out] end;
```

where transmitter is a variable type "processname" which assumes for value the caller's name, [] denotes an empty text message and, therefore, of pure synchronization, and [x] denotes a possibly empty text-message.

On the basis of the above equivalences it is possible to extend the MRD strategy to synchronous-asynchronous communications (SA-MRD strategy) and in this case, the domino effect will be excluded from the system.

Thus, we can say that a Call operation, which is equivalent to an SR sequence, generally needs the insertion of a mark for the MRD strategy.

As regards the Accept command, which is equivalent to an RS sequence, this requires a mark insertion in the MRD strategy if Accept follows a transmission operation or a D length sequence of receiving primitives.

Fig.3 also shows an extension of the Mealy-like FSM of the MRD strategy in Fig.2 to synchronous environment.

4. EVALUATION OF THE MRD RECOVERY POINT INSERTION STRATEGY

Several methods for evaluating rollback recovery systems have been proposed; however, most of these approaches [Chandy75] [Gelembe78] either assume that state restoration can be obtained by a single recovery point or do not include the rollback propagation in other processes. Elsewhere, rollback propagation is taken into account [Lee84]. In this paper, the performance of the MRD marking strategy is evaluated by means of a simulation model; elsewhere [Cantone86], we have set up a mathematical model to evaluate MRD strategy; in the present stage of our research we are comparing results of evaluation and simulation models and hope to publish the results in the near future.

The model that we have developed is, in general, valid both for software faults and for transient or permanent hardware faults. In order to simplify this exposition, we will from now on, refer only to transient faults; furthermore, for the same reasons, we will suppose that the processes can communicate only by means of asynchronous primitives.

In the following sub-sections 4.1 and 4.2 we will discuss the processing system model and the techniques for the treatment of transient faults.

4.1 Processing Sistem Model

In order to evaluate the MRD strategy we have hypothesized that

- a task is formed of a set of NP processes each communicating with other by means of asynchronous message passing;
- process communication rates are high;
- process execution time is similar to total task execution time;
- task termination is conditioned by the termination of its processes;
- each process is executed on a separate processor.

4.2 Treatment of Faults

The treatment of transient faults is carried out in the following two steps:

- error detection;
- rollback to a consistent and error free state.

Now let's make some assumptions on the implementation of these actions:

Assumption 1: each process is partitioned into one or more segments: the tail of each segment consists of a control test, AT, similar to Acceptance Test proposed in [Randell75].

Assumption 2: the test of a process segment (PS) detects every error caused by transient faults occurring in the processor where the PS is running. If the test is successful then the actual state of the process is recovered and memorized in a Base recovery point.

Assumption 3: tests are realiable.

Assumption 4: if the test is unsuccessful then a Kernel procedure stops the normal activity and starts the recovery procedure.

Assumption 5: if an error is detected again during the replay of a PS and this error is the same as the error detected previously in PS, then it is due to a propagation effect.
Assumption 6: each process has a " message history".
Assumption 7: in order to avoid the domino effect, System recovery points are inserted in the process in accordance with the MRD strategy.
Assumption 3 is reasonable, provided that the conditional probability that a transient fault occurs during PS acceptance testing when another fault has been detected during the PS processing, is low.
Assumption 5 is justified by the fact that messages may carry errors; moreover, it also permits the avoidance of deadlock. For example, let’s consider processes P1 and P2, which are described below; test A in process P1 may detect an error, propagated from process P2 by its message m1, earlier than test B of P2 detects it; moreover, if assumption 5 did not hold true, should there be an error in m1, the process P2 would never be able to reach test B, but would, on contrary, be blocked while waiting for the arrival of message m2 from process P1, which would, in turn, be subjected to rollback due to the inevitable failure of its test A.

```
process P1;                         processP2;
 ......                              ......
 receive(p2,m1)                      send(p1,m1)
 ......                              ......
 acceptance_test(A)                  ......
 ......                              receive(p1,m2)
 send(p2,m2)                         ......
 ......                              acceptance_test(B)
 ......                              ......
```

Assumption 2 and assumption 7 give criteria for the insertion of recovery points which may be schematized by an FSM as in Fig.4 where it is hypothesized that the maximum length allowed for a sequence of Receive between two consecutive marks (SRP and/or BRP) is three.
When an error generated by a transient fault is detected by an Acceptance Test , it is necessary to restart the process from a consistent and error-free state. In due course, we shall establish a rule which is useful in reaching this goal. To this end let’s make the following notations:

- PS(p,s), is the segment s-th of process p;
- Rec[PS(p,s)], is the set of receive primitives executed by PS(p,s);
- Send[PS(p,s)], is the set of send primitives executed by PS(p,s);
- RP(r),is the last recovery point (BRP or SRP) inserted in a process before the execution of the receive primitive identified by r.
- BRP(PS(p,s)), is the base recovery point associated to s-th segment of process p.

Definition 1: Let x and y be Send or Receive primitives, we define the subsequent relation T between x and y, noted as x T y (note that T is a restriction of the P relation defined in [Russell80]) when either:
- x and y are in the same process and x occurs before y; or
- x is a Send primitive and y is the associated receive

primitive.

Definition 2: T* is the reflexive transitive closure of T.

Definition 3: The set of process segments which can be called by a PS(p,s), shown by SPSC[PS(p,s)], is the set of all PS(h,k) such that x∈ Send[PS(p,s)], y∈ Rec[PS(h,k)]: x T* y.

When an error is detected in a PS(p,s), the recovery procedure can be implemented in accordance with the following rule:

Rule 1: in case of error detection in a PS(p,s), it is necessary to i) restore the p computation into the associated state of BRP(PS(p,s)); ii) revoke all the y∈ Rec[PS(h,k)] primitives which are in the relation x T* y, where x∈ Send[PS(p,s)]; iii) restore all the PS(h,k)∈ SPSC[PS(p,s)] into the state associated to the Mark(r), where r is the first receive revoked in step ii).

4.3 Guidelines for the Evaluation of the MRD Strategy

In order to determine the mean of the time, Treal, necessary to execute a task in a processing environment such as that described in 4.1 and in which features are provided in order to deal with fault occurrences as described in 4.2 we have used a simulation model.

In particular, we have hypothesized that each task process is executed on an individual NP processor; each processor may be subjected to faults, and occurrences of faults can be modelled on a Poisson process with fault-rate equal to Lfault.

When a j-th fault occurs and it is detected, for example, by the Acceptance Test of PS(p,s), then the recovery (cfr. Rule 1) is propagated to all PS(h,k)∈ SPSC[PS(p,s)].

For example, in Fig.5, where the SRPs are inserted in such a way that D=3, should a fault be detected at point tf in the process P2, the process that would lose most computation due to rollback is process P4: its computation lost would be equal to E[Tps]+2*E[Trr], where Tps is the expected execution time for a process segment in error-free condition and Trr is the expected time between two consecutive receives.

Moreover, in the simulation model we have hypothesized that i) the time between two consecutive receives or send primitives, respectively, ii) the duration of a process segment, iii) the duration of a process, iv) the time necessary to execute an acceptance test, v) the times necessary to insert a Base or a System Recovery Point are all random variables with a probability density function of exponential type with an exponent parameter equal, to Lrec, Lsnd, Lnew, Lend, Lact, Lbrp, Lsrp, respectively.

5. CONCLUSIONS AND FINAL REMARKS

Fig.6 shows the shape of E[Treal] in function of D which was obtained from the simulation of a task formed of six twin processes, having Lrec=Lsnd.

Fig.7 shows the shape of E[Treal] in function of D that was obtained from the simulation of a task formed of six processes, of which three are producer-like (send-bound) processes and three are consumer-like (Receive-bound) processes. By refering to Fig.6 it can be seen that there is a threshold (Dt) which is such that, starting from D=1, as D increases, the value of E[Treal] decreases and then, starting from D=Dt, becomes practically constant. This is congruent with our hypotheses: indeed, on the left side of Dt, it is more than likely that, since D is very small, then is very easy to obtain dynamic Receive sequences of not less than D in length, and that, therefore, there will be a considerable overhead, due to the regular insertion of system recovery points, one for every dynamic receive sequence of not less than D in length. This overhead decrease as D increases but with an ever diminisming differential. This is due to the fact that as D increases the probability of there being receive sequences less than D in length decreases.

In Fig.7, on the other hand, the minimum value point, Dm, for E[Treal](D) can be seen, and this is due to the presence of receive-bound processes. This depends on the fact that, because of the type of processes involved, the probability of their being dynamic Receive sequences with a length not less than D is very high, when compared to that of previous case. Indeed, for low values of D the overhead associated with the dynamic insertion of SRPs is high while computation lost should rollback occur is low. For high values of D, hovewer behaviour reciprocal to those described above are obtained. Note that this interpretation of the results is correborated by the fact that i) Dm depends inverserly on Lfault; ii) the slope of the curves increases on the right of Dm when Lfault increases.

As far as possible future developments of this work are concerned, we are working towards comprehensively defining the mathematical model of the strategy put forward by following similar lines to those used in [Lee84] for the evaluation of hardware recovery blocks. The results referring to an early version of this model are available from the authors.

6. REFERENCES

[Cantone86] G. Cantone, B. Ciciani: A Mathematical Model to Evaluate a New Optimizing Recovery Point Insertion Strategy for Distributed Computing Systems. Unpublished Int. Rep., 1986

[Chandy85] K.M. Chandy, J.C. Browne, C.W. Display, W.R. Uhrig: Analytic Models for Rollback and Recovery Strategies in Data-Base Systems. IEEE Tr. on Soft. Eng., Vol. SE-1, March 1985.

[Cristian85] F. Cristian: A Rigorous Approach to Fault-Tolerant Programming. IEEE Trans. on Software Engineering, Vol. SE-11, No. 1, January 1985

[Gelembe78] E. Gelembe, D. Derochette: Performance of Rolback Recovery Systems under Intermittent Failures. Comm. of ACM, Vol.21, June 1978.

[Lee84] Y-H. Lee and K.G. Shin: Design and Evaluation of a Fault-Tolerant Multiprocessor Using Hardware Recovery Blocks. IEEE Tr. on Computers, Vol. C-33, n.2, February 1984

[Randell75] B. Randell: System Structure for Software Fault-Tolerance. IEEE Trans. on Software Engineering, Vol.SE-1, n.2, June 1975.

[Russell80] D.L.Russell: State Restoration in Systems of Communicating Processes. IEEE Tr. on Soft. Eng. Vol.SE-6, n.2, March 1980.

[Shin84] K.G. Shin, Y-H. Lee: Evaluation of Error Recovery Block used for Cooperating Processes. IEEE Tr. on Soft. Eng., Vol. SE-10. No.6, November 1984.

[Velardi83] P. Velardi, B. Ciciani: Recovery Blocks for Communicating Systems. Microprocessing and Microprogramming, No. 11, Springer-Verlag, 1983.

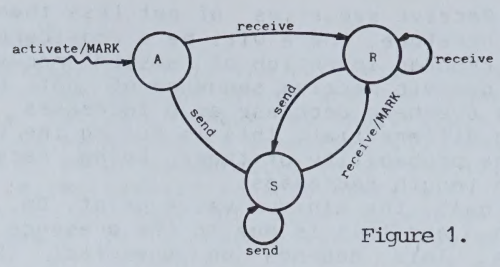

activete/MARK
receive
A
R
receive
send
send
receive/MARK
S
send

Figure 1.

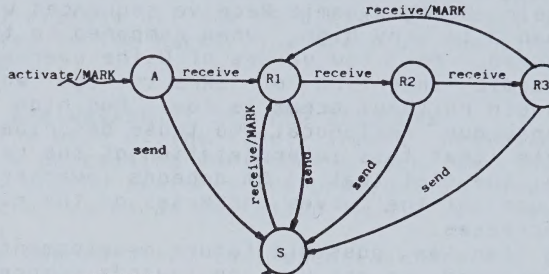

receive/MARK
activate/MARK
A
receive
R1
receive
R2
receive
R3
send
receive/MARK
send
send
send
S
send

Figure 2.

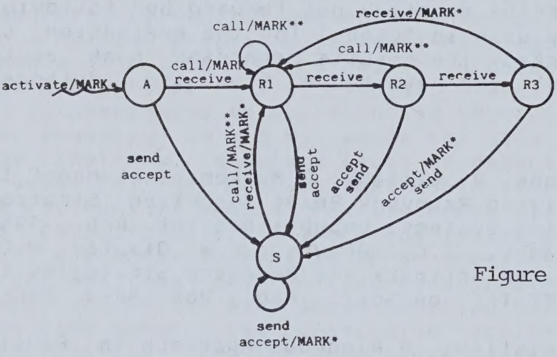

call/MARK**
receive/MARK*
call/MARK**
call/MARK**
call/MARK
receive
activate/MARK
A
R1
receive
R2
receive
R3
send
accept
call/MARK**
receive/MARK*
send
accept
accept
send
accept/MARK*
send
S
send
accept/MARK*

Figure 3.

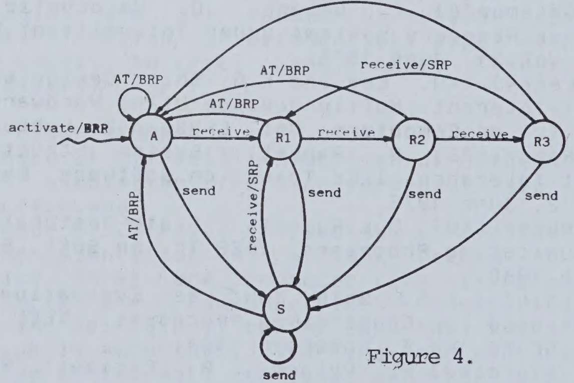

AT/BRP
AT/BRP
receive/SRP
AT/BRP
AT/BRP
activate/BRP
A
receive
R1
receive
R2
receive
R3
AT/BRP
send
receive/SRP
send
send
send
S
send

Figure 4.

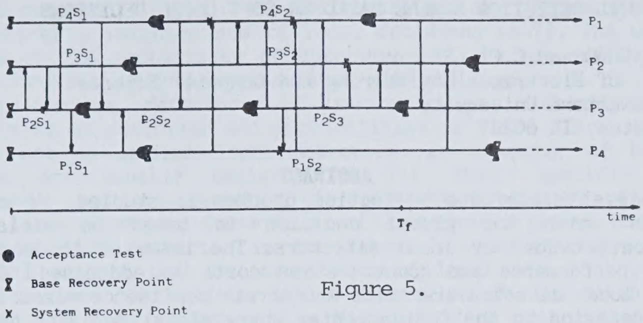

P4S1
P4S2
P1
P3S1
P3S2
P2
P3
P2S1
P2S2
P2S3
P4
P1S1
P1S2
Tr
time
Acceptance Test
Base Recovery Point
System Recovery Point

Figure 5.

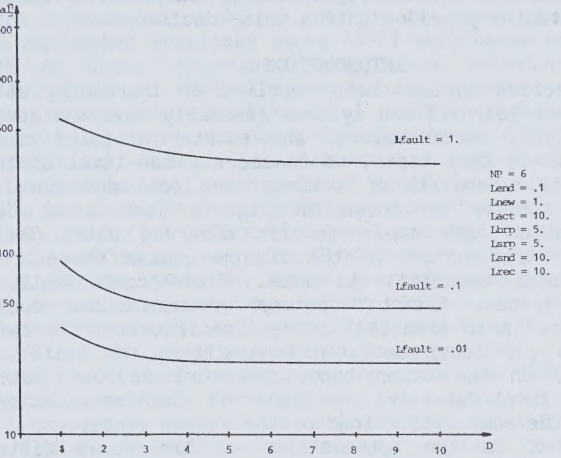


Figure 6.

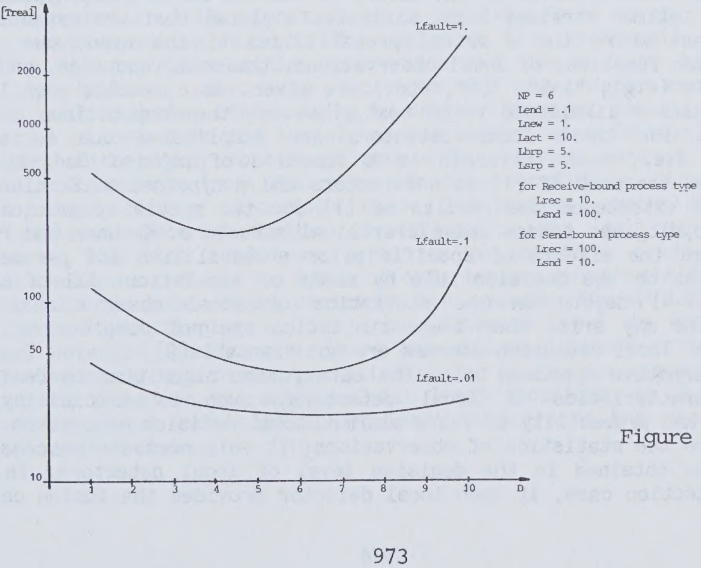


Figure 7.

A DISTRIBUTED DETECTION SCHEME BASED ON SOFT LOCAL DECISIONS

J.J. CHAO and C.C. LEE
Dept. of Electrical Engineering and Computer Science
Northwestern University
Evanston, IL 60201

ABSTRACT

A two-level distributed detection problem is studied. We consider a system in which the global decision is based on decision-level information provided by local detectors. The issue of tradeoff between detection performance and communication costs is addressed. We propose that each local detector transmits a discrete confidence measure together with its decision to the fusion center where global decision is made. We showed that the proposed system significantly outperforms the one in which each local detector provides only a crisp decision.

1. INTRODUCTION

Distributed detection systems have received an increasing attention over the past several years. These systems generally have the advantages of higher reliability, survivability, and shorter decision time than centralized systems. In this paper, we consider a two-level distributed detection system which consists of a number of local detectors and an information fusion center as shown in Fig. 1. Each local detector performs the detection task based on its observed data. The local decisions are then transmitted to the fusion center where a global decision (i.e., final decision) is made. Since communications are involved in the system, tradeoff between communication costs and detection performance is an essential issue. Specifically, the larger is the number of bits per local decision transmitted, the better is the detection accuracy. On the other hand, the more is the information transmitted by each local detector, the higher is the communication costs and the heavier is the computation load of the fusion center.

Several approaches to the optimization of the above distributed detection system have been discussed in the literature [1-8]. Tenney [1] extended the classical detection theory to the two-sensor, two-hypothesis test. The optimum strategy that minimizes a global cost was established in the case where the a priori probabilities of the hypotheses, the distribution functions of local observations, the cost functions, and the fusion rule (e.g., "AND", "OR", etc.) are given. As a result, each local detector uses a likelihood ratio test. However, the computations of the thresholds for these local detectors are complicated due to cross coupling, i.e., each threshold is a function of others. Sadjadi [4] generalized the work of [1] to n detectors and m hypotheses. Ekchian and Tenney [3] extenteded the results of [1] for two special communication network topologies: tandem and hierarchical structure. Kushner and Pacut [2] studied the effects of specific prior probabilities and parametric dependencies on the decision rule by means of simulation. All of these studies [1-4] depend on the statistics of local observations, and difficulties may arise when these statistics are not complete or when analysis of local detection schemes are not tractable[9].

An alternative approach bases the data fusion algorithms on decision level characteristics of local detectors, such as probability of detection and probability of false alarm. Global decision maker then does not require the statistics of observations; it only needs to process the information obtained in the decision level of local detectors. In the binary detection case, if each local detector provides the fusion center a one-bit decision, (e.g., crisp decision) then the optimum decision rule is in general a weighted sum of local decisions [5-7]. The weights are functions of the probability of detection and the probability of false alarm. A crucial advantage associated with this approach is that it is easy to combine detectors of various types, which have known probabilities of detection and probabilities of false alarm, even though they may not be optimum local detectors. As a matter of fact, local detectors are usually designed to fit their specific detection environments, and are possibly of different types. By means of the decision level approach, Conte et al. [5] derived a global decision rule based on an "OR" criterion in the Neyman-Pearson strategy. Chair and Varshney[6] assumed that decision rules of local detectors were known, and then derived the optimum fusion algorithms in the Bayesian sense. Srinivasan [7] also worked on the Neyman-Pearson distributed detection problems and indicated that the optimum decision threshold of a local detector depends on the fusion rule and on other local detectors.

The approaches mentioned above [5-7] were based on the 1-bit decision results of local detectors. Each local detector makes a definite decision: "Yes" or "No". This certainly minimizes communication costs. However, to adequately utilize local decision information, it may not be necessary to restrict the local decision to be a "crisp decision" (i.e., 1-bit decision). Indeed, just like what a sign test does to the testing of a location shift, a crisp local decision means a lot of information loss. In practice, the communication channel may be capable of handling more than one bit of information per local decision. Then, in addition to the crisp decision, each local detector can transmit a "confidence measure" of its decision. These confidence measures would assist the fusion center in assigning proper weights to local decisions. The confidence measure has to be quantized according to channel capacity, of course. Since the local decision is no longer just a crisp result in this case, the traditional logic operations, such as "AND", "OR", "MAJORITY", etc., are not suitable for combining local decisions.

We propose that every local detector send multi-bit information to the fusion center. We address the issue of optimum partition of a decision space with respect to a local detector. In the "crisp decision" case, the decision space is partitioned into exclusive regions, each corresponding to a hypothesis. A decision is made in complete favor of one hypothesis, when the observation falls in the region of this hypothesis, regardless of the distance between the observation and the decision boundary. In reality, however, some local detectors may have higher confidence on their decisions than others owing to certain physical situations such as relative distances to the target, aspect angles, etc. Certainly, a local decision with higher confidence should be weighted more in the global decision than is a local decision with lower confidence. Hence, instead of a crisp partition of decision space, the proposed system entails that the decision space be further partitioned in the light of confidence measures. With the confidence partition on top of a crisp partition, local decisions provide two kinds of information: hypothesis decision and decision confidence. These decisions are somewhat "fuzzy". Clearly, the confidence partitioning at the local detector level is essential for global detection performance. Using the maximum distance criterion [10], we derive an optimum binary partitioning for confidence measures of local detectors. Based on the optimum 2-bit local decisions, we show that the resulting global detector significantly outperforms the one that uses crisp local decisions.

2. A DISTRIBUTED DETECTION SCHEME BASED ON SOFT LOCAL DECISIONS

We consider a distributed detection system which has n independent local detectors. For every local detector, we assume that the functional relationship between the false alarm probability and the probability of detection are known and that the a priori probabilities of the two hypotheses H and K are not available. Let $(x_1, x_2, \ldots, x_k)$ be the observation vector of a local detector. Then the local decision space is given by $\Omega = \{(x_1, x_2, \ldots, x_k) |$ all possible realizations of x_j, $j=1,2,\ldots,k \}$. If a local decision u_i is crisp, then the associated decision space Ω is partitioned into two exclusive regions, Ω_H and Ω_K, such that

$$u_i = H, \quad \text{if} \quad (x_1,x_2,\ldots,x_k) \in \Omega_H$$
$$u_i = K, \quad \text{if} \quad (x_1,x_2,\ldots,x_k) \in \Omega_K$$

The above decision rule is associated with a probability of false alarm:

$$P\{ u_i=K \mid H \}=a_i, \quad i=1,2,\ldots,n \tag{2-1}$$

and a probability of detection:

$$P\{ u_i=K \mid K \}=b_i, \quad i=1,2,\ldots,n. \tag{2-2}$$

Both a_i and b_i depend on the detection threshold whose optimum value with respect to the global decision can be determined [7]. In order to improve the performance, every crisp decision can be generalized to include a confidence measure. Specifically, in the decision space, each of Ω_H and Ω_K can be further partitioned into m subsets, respectively. In the case of m=2, we have

$$\Omega_H = \Omega_{H_1} \cup \Omega_{H_2}, \qquad \Omega_{H_1} \cap \Omega_{H_2} = \emptyset$$
$$\Omega_K = \Omega_{K_1} \cup \Omega_{K_2}, \qquad \Omega_{K_1} \cap \Omega_{K_2} = \emptyset .$$

The associated decision rule is then given by

$u_i = H_1$ (decision is "H" with lower confidence), if $(x_1,x_2,\ldots,x_k) \in \Omega_{H_1}$;

$u_i = H_2$ (decision is "H" with higher confidence), if $(x_1,x_2,\ldots,x_k) \in \Omega_{H_2}$;

$u_i = K_1$ (decision is "K" with lower confidence), if $(x_1,x_2,\ldots,x_k) \in \Omega_{K_1}$;

$u_i = K_2$ (decision is "K" with higher confidence), if $(x_1,x_2,\ldots,x_k) \in \Omega_{K_2}$.

Corresponding to a_i and b_i in the crisp partition case, we have, for $1 \leq i \leq n$, $j=1,2$,

$$P\{ u_i=K_j \mid H \}=a_{ij}; \qquad P\{ u_i=H_j \mid H \}=p_{ij};$$
$$P\{ u_i=K_j \mid K \}=b_{ij} \quad \text{and} \quad P\{ u_i=H_j \mid K \}=q_{ij}. \tag{2-3}$$

These numbers can be regarded as possible confidence assignments of local decision u_i. They satisfy

$$a_{i1}+a_{i2}=a_i; \qquad b_{i1}+b_{i2}=b_i;$$
$$p_{i1}+p_{i2}=p_i = 1-a_i \quad \text{and} \quad q_{i1}+q_{i2}=q_i = 1-b_i. \tag{2-4}$$

Let $\underline{U}=\{u_1,u_2,\ldots,u_n\}$ be local decisions which form the observation vector of the global decision maker. Assuming independence among local detectors, the likelihood ratio is given by

$$\frac{P(\underline{U}|K)}{P(\underline{U}|H)} = \frac{P(u_1,u_2,\ldots,u_n|K)}{P(u_1,u_2,\ldots,u_n|H)} = \prod_{i=1}^{n} \frac{P(u_i|K)}{P(u_i|H)}$$

$$= \prod_{i=1}^{n_{K_1}} \frac{P(u_i=K_1|K)}{P(u_i=K_1|H)} \prod_{i=1}^{n_{K_2}} \frac{P(u_i=K_2|K)}{P(u_i=K_2|H)} \prod_{i=1}^{n_{H_1}} \frac{P(u_i=H_1|K)}{P(u_i=H_1|H)}$$

$$\cdot \prod_{i=1}^{n_{H_2}} \frac{P(u_i=H_2|K)}{P(u_i=H_2|H)} \tag{2-5}$$

where n_{Y_j} is the number of local decisions u_i which assume Y_j, Y=H,K, j=1,2, and $n_{K_1}+n_{K_2}+n_{H_1}+n_{H_2}=n$. The Neyman-Pearson test follows:

$$T(\underline{U}) = \log \frac{P(\underline{U}|K)}{P(\underline{U}|H)} \begin{matrix} > \\ < \end{matrix} t \quad \begin{matrix} \Rightarrow K \\ \Rightarrow H \end{matrix} \tag{2-6}$$

where t is the threshold of the test. The test statistic $T(\underline{U})$ can be written as

$$T(\underline{U}) = \log \frac{P(\underline{U}|K)}{P(\underline{U}|H)} = \sum_{i=1}^{n_{K_1}} \log \frac{P(u_i=K_1|K)}{P(u_i=K_1|H)} + \sum_{i=1}^{n_{K_2}} \log \frac{P(u_i=K_2|K)}{P(u_i=K_2|H)}$$

$$+ \sum_{i=1}^{n_{H_1}} \log \frac{P(u_i=H_1|K)}{P(u_i=H_1|H)} + \sum_{i=1}^{n_{H_2}} \log \frac{P(u_i=H_2|K)}{P(u_i=H_2|H)}$$

$$= \sum_{i=1}^{n_{K_1}} \log \frac{b_{i1}}{a_{i1}} + \sum_{i=1}^{n_{K_2}} \log \frac{b_{i2}}{a_{i2}} + \sum_{i=1}^{n_{H_1}} \log \frac{q_{i1}}{p_{i1}} + \sum_{i=1}^{n_{H_2}} \log \frac{q_{i2}}{p_{i2}}$$

$$\triangleq \sum_{i=1}^{n} w_i \tag{2-7}$$

$$\text{where } w_i = \begin{cases} \log \dfrac{b_{ij}}{a_{ij}} & \text{if } u_i=K_j \\ \log \dfrac{q_{ij}}{p_{ij}} & \text{if } u_i=H_j \end{cases} \tag{2-8}$$

$i=1,2,\ldots,n, \quad j=1,2.$

The test statistic is a weighted sum of local decisions; the weights being functions of confidence assignments. We assume that a_i and b_i are given. In other words, the decision boundary between Ω_H $(= \Omega_{H_1} \cup \Omega_{H_2})$ and Ω_K $(= \Omega_{K_1} \cup \Omega_{K_2})$ is fixed. We will study the partitioning of Ω_H and Ω_K by means of partitioning a_i and p_i. Due to the complicated functional form of the probabilities of error, we employ the maximum distance criterion [10-14] for optimization. The optimum partitioning of Ω_H and Ω_K can be obtained using this criterion. Specifically, J-Divergence [11] is used as a performance index. The J-Divergence can be expressed as

$J = E_K[T(\underline{U})] - E_H[T(\underline{U})]$,

where $E_Y[\cdot]$ is the expectation taken under hypothesis Y, Y=H,K. It follows that

$$J = E_K\left[\sum_{i=1}^{n} w_i\right] - E_H\left[\sum_{i=1}^{n} w_i\right] = \sum_{i=1}^{n} (E_K[w_i]-E_H[w_i]) \triangleq \sum_{i=1}^{n} J_i \tag{2-9}$$

where $J_i = E_K[w_i] - E_H[w_i]$, are local J-divergences. Hence, the partition of each local decision space is independent of those of others in the sense of J-divergence. We have

$$
\begin{aligned}
J_i &= E_K[w_i] - E_H[w_i] \\
&= \left[\, q_{i1} \log\frac{q_{i1}}{p_{i1}} + q_{i2} \log\frac{q_{i2}}{p_{i2}} + b_{i1} \log\frac{b_{i1}}{a_{i1}} + b_{i2} \log\frac{b_{i2}}{a_{i2}} \,\right] \\
&\quad -\left[\, p_{i1} \log\frac{q_{i1}}{p_{i1}} + p_{i2} \log\frac{q_{i2}}{p_{i2}} + a_{i1} \log\frac{b_{i1}}{a_{i1}} + a_{i2} \log\frac{b_{i2}}{a_{i2}} \,\right] \\
&= (q_{i1}-p_{i1}) \log\frac{q_{i1}}{p_{i1}} + (q_{i2}-p_{i2}) \log\frac{q_{i2}}{p_{i2}} + (b_{i1}-a_{i1}) \log\frac{b_{i1}}{a_{i1}} \\
&\quad +(b_{i2}-a_{i2}) \log\frac{b_{i2}}{a_{i2}} \qquad (2\text{-}10)
\end{aligned}
$$

Assume that the functional relationship between a_i, b_i is given (Fig. 2a) for each local detector, namely, $b_i=f(a_i)$. Then, as is described in Fig. 2b, the functional relationship of b_{i2} and a_{i2} is known, i.e., $b_{i2}=f(a_{i2})$. Thus we may write

$$
\begin{aligned}
a_{i1} &= a_i - a_{i2} \\
b_{i1} &= b_i - b_{i2} = b_i - f(a_{i2}). \qquad (2\text{-}11)
\end{aligned}
$$

Then, given a_i, b_i, we may proceed to find optimum value of a_{i2} from which optimum a_{i1}, b_{i1}, and b_{i2} are determined.

Differentiating J_i with respect to a_{i2}, we obtain

$$
\begin{aligned}
\frac{\partial J_i}{\partial a_{i2}} &= \frac{\partial}{\partial a_{i2}} \left\{ [\,(b_i-b_{i2})-(a_i-a_{i2})] \log\frac{b_i-b_{i2}}{a_i-a_{i2}} \right. \\
&\qquad \left. +(b_{i2}-a_{i2}) \log\frac{b_{i2}}{a_{i2}} \right\} \\
&= f'(a_{i2}) \left[-\frac{b_i-a_i}{b_i-f(a_{i2})} - \log\frac{b_i-f(a_{i2})}{a_i-a_{i2}} + \frac{f(a_{i2})-a_{i2}}{b_i-f(a_{i2})} \right. \\
&\qquad \left. + \log\frac{f(a_{i2})}{a_{i2}} + \frac{f(a_{i2})-a_{i2}}{f(a_{i2})} \right] + \frac{b_i-a_i}{a_i-a_{i2}} \\
&\qquad + \log\frac{b_i-f(a_{i2})}{a_i-a_{i2}} - \frac{f(a_{i2})-a_{i2}}{a_i-a_{i2}} - \log\frac{f(a_{i2})}{a_{i2}} \\
&\qquad - \frac{f(a_{i2})-a_{i2}}{a_{i2}} \qquad (2\text{-}12)
\end{aligned}
$$

where $f'(.)$ is the first derivative of $f(\cdot)$. Setting

$$\frac{\partial J_i}{\partial a_{i2}} = 0 \qquad \text{yields}$$

$$f'(a_{i2})=\frac{\dfrac{b_i-a_i}{a_i-a_{i2}}+\log\dfrac{b_i-f(a_{i2})}{a_i-a_{i2}}-\dfrac{f(a_{i2})-a_{i2}}{a_i-a_{i2}}-\log\dfrac{f(a_{i2})}{a_{i2}}-\dfrac{f(a_{i2})-a_{i2}}{a_{i2}}}{\dfrac{b_i-a_i}{b_i-f(a_{i2})}+\log\dfrac{b_i-f(a_{i2})}{a_i-a_{i2}}-\dfrac{f(a_{i2})-a_{i2}}{b_i-f(a_{i2})}-\log\dfrac{f(a_{i2})}{a_{i2}}-\dfrac{f(a_{i2})-a_{i2}}{f(a_{i2})}} \qquad (2\text{-}13)$$

Its solution gives the optimum partition of a_i, and b_i.

Similarly, as is illustrated in Fig. 2c,

we have
$$\begin{aligned} q_{i1}&=f(a_i+p_{i1})-b_i \\ q_{i2}&=1-b_i-q_{i1}=1-f(a_i+p_{i1}) \\ p_{i2}&=1-a_i-p_{i1}. \end{aligned} \qquad (2\text{-}14)$$

Setting $\dfrac{\partial J_i}{\partial p_{i1}}=0$, and using $s_i=a_i+p_{i1}$ yields

$$f'(s_i)=\frac{\log\dfrac{1-f(s_i)}{1-s_i}+\dfrac{[1-f(s_i)]-(1-s_i)}{1-s_i}-\log\dfrac{f(s_i)-b_i}{p_{i1}}-\dfrac{f(s_i)-b_i-p_{i1}}{p_{i1}}}{\log\dfrac{1-f(s_i)}{1-s_i}+\dfrac{[1-f(s_i)]-(1-s_i)}{1-f(s_i)}-\log\dfrac{f(s_i)-b_i}{p_{i1}}-\dfrac{f(s_i)-b_i-p_{i1}}{f(s_i)-b_i}} \qquad (2\text{-}15)$$

We can thus obtain the optimum partitioning of p_i.

3. PERFORMANCE

Let $A=\{(u_1,\cdots,u_n);\ \sum_{i=1}^{n} w_i \geqslant t\}$, then the false alarm probability is

$$P_F=\sum_{\underline{u}\epsilon A} P(\underline{u}|H)=\sum_{\underline{u}\epsilon A}\Big[\prod_{i=1}^{n} P(u_i|H)\Big], \qquad (3\text{-}1)$$

and the detection probability is

$$P_D=\sum_{\underline{u}\epsilon A} P(\underline{u}|K)=\sum_{\underline{u}\epsilon A}\Big[\prod_{i=1}^{n} P(u_i|K)\Big]. \qquad (3\text{-}2)$$

As an example, we consider the problem of radar detection under the environment of Rayleigh fluctuation. We assume that all local detectors are identical Neyman-Pearson optimum detectors (i.e., matched filter followed by envelope detector) and that all have the same signal to noise ratio. In this case, for every local detector, the relationship between the probability of detection b and the probability of false alarm a is known as [15]

$$b=(a)^{1/(1+r)}$$

where r is the average signal energy to noise ratio. Thus we have

$$f(x)=x^{1/(1+r)} \qquad (3\text{-}3)$$

and

$$f'(x)=(1/1+r)x^{-r/(1+r)} \qquad (3\text{-}4)$$

for Equations (2-13) and (2-15). First we assume that a is fixed. Then, for a given r, we can calculate b and use (3-3) and (3-4) to solve (2-13) and (2-15) and thereby obtain optimum values for a_2 and p_1. Note that we have dropped the subscript i since all local detectors are identical. Using a_2 and p_1, we can find a_1, b_1, b_2, p_2, q_1, and q_2 from Equations (2-4), (2-14), and (3-3). Then, the values of w_i given a local decision u_i are determined (Eq.(2-8)). Finally, P_F and P_D are readily computed from (3-1) and (3-2). Conversely, if P_F is fixed for all r, then for each value of r, we can compute the value of a that satisfies P_F provided that optimum partitionings of a and p are employed. In Figure 3, we compare performances of the optimum centralized detection system, the distributed detection system using crisp local decisions and the proposed distributed detection system using 2-bit fuzzy local decisions. Each system has ten identical (optimum) local detectors and uses a P_F=0.01. The optimum centralized detection system bases it decision on incoherent integration of local "observations" (i.e., Swerling II). It shows that the distributed system using 2-bit local decisions significantly outperforms the one using crisp local decisions and is close to the optimum centralized system. Figure 4 compares only two distributed systems each of which has twenty local detectors. The performance improvement by using one extra bit per decision is even more significant in this case.

4. CONCLUDING REMARKS

We have studied a two-level distributed detection system in which the global decision is based on decision-level information of local detectors. Instead of using crisp local decisions alone in global decision making, we propose to use confidence measure on top of local decisions. This entails that each local detector transmit multiple bits per decision to the fusion center. Using the maximum distance criterion, we were able to optimize a "one bit" confidence information. As a result, we show by an example that the detection performance is significantly improved from the case where crisp local decisions are used.

Generalizations of the results to the case of multiple-bit confidence information is possible but will be more difficult. Essentially, the functional relationship between partitioned false alarm probabilities and detection probabilities described in Figure 2 will become inapplicable. It is also possible to study the optimum partitioning problem using criteria other than J-divergence. For example, Matsusita's distance [14] and Bhattacharyya's distance [11] can be considered.

ACKNOWLEDGEMENT

This research was partially supported by a research grant from Recon Optics, Inc., Barrington, Illinois.

REFERENCES

1. R.R. Tenney and N.R. Sandell,Jr., "Detection With Distributed Sensors," IEEE Trans. Aerospace and Electronic Systems, vol. AES-17, pp. 501-510, July 1981.
2. H.J. Kushner and A. Pacut, "A Simulation Study of Decentralized Detection Problem," IEEE Trans. Automatic Control, vol. AC-27, pp. 1116-1119, Oct 1982.
3. L.K. Ekchian and R.R. Tenney, "Detection Networks," Proceedings of the 21st IEEE Conference on Decision and Control, Fla., pp. 686-691, Dec. 1982.
4. F. Sadjadi, "Hypotheses Testing in a Distributed Enviroment," IEEE Trans. Aerospace and Electronic Systems, vol. AES-22, pp. 134-137, Mar. 1986.

5. E. Conte et al., "Multistatic Radar Detection: Synthesis and Comparision of Optimum and Suboptimum Receivers," IEE Proceedings, vol. 130, Pt. F, pp. 484-494, Oct. 1983.
6. Z. Chair and P.K. Varshney, "Optimum Data Fusion in Multiple Sensor Detection Systems," IEEE Trans. Aerospace and Electronic Systems, vol. AES-22, pp. 98-101, Jan. 1986.
7. R. Srinivasan, "Distributed Radar Detection Theory," IEE Proceedings, vol. 133, Pt. F, pp. 55-60, Feb. 1986.
8. J.N. Tsitsiklis and M. Athans, "On the complexity of Decentralized Decision Making and Detection problems," IEEE Trans. Automatic Control, vol. AC-30, pp. 440-446, May 1985.
9. C.C. Lee and J.B. Thomas, "Detectors for Multinomial Input," IEEE Trans. Aerospace and Electronic Systems, vol. AES-19, pp. 288-297, Mar. 1983.
10. T.L. Grettenberg, "Signal Selection in Communication and Radar Systems," IEEE Trans. Information Theory, vol. IT-9, pp.265-275, Oct. 1963.
11. T. Kailath, " The Divergence and Bhattacharyya Distance Measures in Signal Selection," IEEE Trans. Comm. Tech., vol. COM-15, pp. 52-60, Feb. 1967.
12. S.M. Ali and S.D. Silvey, "A General Class of Coefficients of Divergence of One Distribution from Another," J. Royal Stat. Soc., Series B,vol. 28, pp. 131-142, 1966.
13. H. Kobayashi, "Distance Measures and Asymptotic Relative Efficiency," IEEE Trans. Information Theory, vol. IT-16, pp. 288-291, May 1970.
14. H.V. Poor and J.B. Thomas, "Applications of Ali-Silvey Distance Measures in the Design of Generalized Quantizers for Binary Decision Systems," IEEE Trans. Communications, vol. COM-25, pp. 893-900, Sep. 1977.
15. J.V. Difranco and W.L. Rubin, "Radar Detection," Prentice-Hall, NJ., 1968

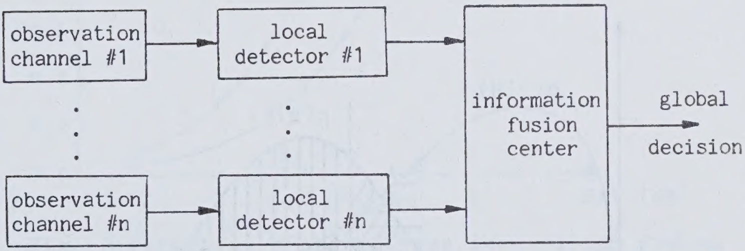


Fig. 1 A two-level distributed detection system.

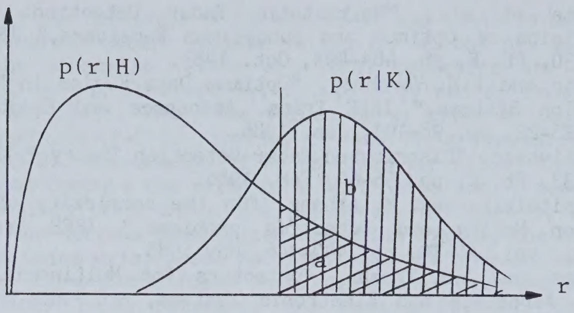


Fig. 2a b = f(a).

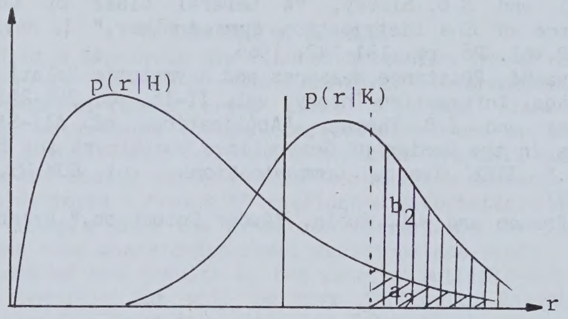


Fig. 2b $b_2 = f(a_2)$.

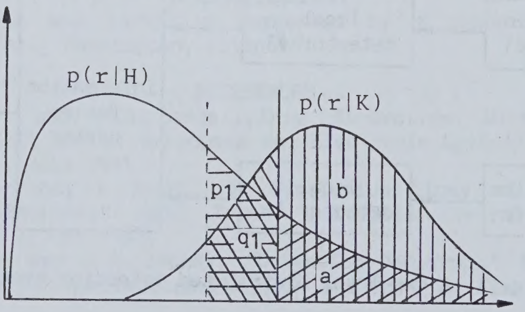


Fig. 2c $b+q_1 = f(a+p_1)$.

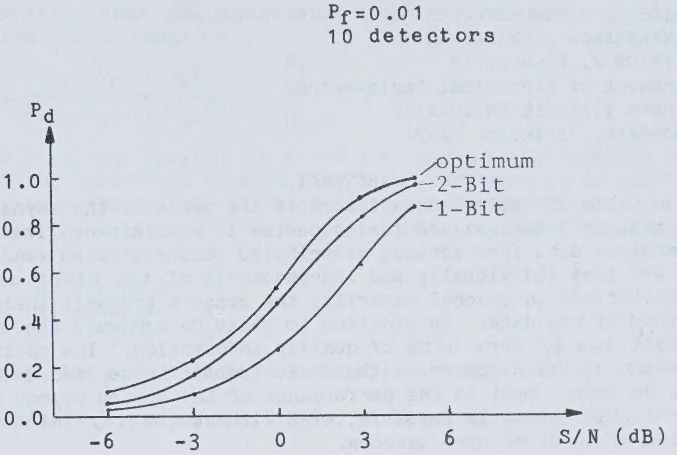


Fig. 3 Detection probabilities in Rayleigh fading with ten local detectors.

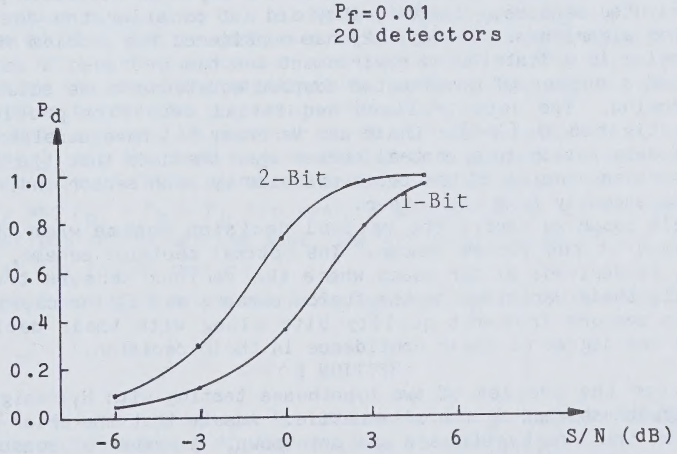


Fig. 4 Detection probabilities in Rayleigh fading with twenty local detectors.

OPTIMAL DECISION FUSION IN MULTIPLE SENSOR SYSTEMS

STELIOS C. A. THOMOPOULOS *
RAMANARAYANAN VISWANATHAN *
DIMITRIOS P. BOUGOULIAS
Department of Electrical Engineering
Southern Illinois University
Carbondale, Illinois 62901

ABSTRACT

The problem of optimal data fusion in the sense of the Neyman-Pearson (N-P) test in a centralized fusion center is considered. The fusion center receives data from various distributed sensors. Each sensor implements a N-P test individually and independently of the other sensors. Due to limitations in channel capacity, the sensors transmit their decisions instead of raw data. In addition to their decisions, the sensors may transmit one or more bits of quality information. The optimal, in the N-P sense, decision scheme at the fusion center is derived and it is seen that an improvement in the performance of the system beyond that of the most reliable sensor is feasible, even without quality information, for a system of three or more sensors.

INTRODUCTION

The problem of data fusion in a central decision center has attracted the attention of several investigators due to the increasing interest in the demployment of multiple sensors for communication and surveillance purposes. A centralized fusion center is responsible for combining the received information from the various sensors into a final decision.

Tenney and Sandell [1] have treated the Bayesian detection problem with distributed sensors. However, they did not consider the design of data fusion algorithms. Sandaji [2] has considered the problem of hypothesis testing in a distributed environment and has provided a solution in terms of a number of complicated coupled equations whose solution is not forthcoming. The decentralized sequential detection problem has been investigated in [3-5]. Chair and Varshney [6] have considered the problem of data fusion in a central center when the data that the fusion center receives consist of the decisions made by each sensor individually and independently from each other.

In this paper we derive the optimal decision scheme when the N-P test is used at the fusion center. The optimal decision scheme, in the N-P sense, is derived: a) for cases where the various sensors transmit exclusively their decisions to the fusion center; and b) for cases where the various sensors transmit quality bits along with their decisions indicating the degree of their confidence in their decision.

SECTION I

Consider the problem of two hypotheses testing with H_1 designating the true hypotheses and H_0 the alternative. Assume that the prior probabilities on the two hypotheses are not known. A number of sensors, N, receive observations and implement the N-P test independently from each other. Let u_j designate the decision of the j-th sensor having taken into account all the observations available to this sensor at the time of the decision. If the decision of the j-th sensor favors hypothesis H_1, the sensor sets $u_j = +1$, otherwise it sets $u_j = -1$ if the decision is in favor of the alternative hypothesis. Every sensor transmits its decision to the fusion center, so that the fusion center has all N decisions available for processing at the time of the decision making. Let (P_{F_j}, P_{D_j}) designate the pair of the probability of false alarm and the

---

*Work supported by the Office of Naval Research under grant N00014-86-K-0515

probability of detection at which the j-th sensor operates and implements the N-P test. The fusion center implements the N-P test using all the decisions that the individual sensors have communicated, i.e. it formulates the LR test:

$$\Lambda(u) = \frac{P(u_1,u_2,\dots,u_N|H_1)}{P(u_1,u_2,\dots,u_N|H_0)} \overset{H_1}{\underset{H_0}{\gtrless}} t \tag{1}$$

where $u = (u_1,u_2,\dots,u_N)$ is a 1xN row vector with entries the decisions of the individual sensors, and t the threshold to be determined by the desirable probability of false alarm at the fusion center, P_F^f.

Since the decisions of each sensor are independent from each other, the LR test (1) gives

$$\Lambda(u) = \prod_{i=1}^{N} \frac{P(u_i|H_1)}{P(u_i|H_0)} \overset{H_1}{\underset{H_0}{\gtrless}} t \tag{2}$$

from which the result in [6] is readily obtained. Thus, it follows from (2):

$$P(\log\Lambda(u)|H_0) = P(\log\Lambda(u_1)|H_0)*\dots*P(\log\Lambda(u_N)|H_0) \tag{3}$$

where

$$P(\log\Lambda(u_i)|H_0) = (1-P_{F_i})\,\delta\left(\Lambda(u_i) - \log\frac{1-P_{D_i}}{1-P_{F_i}}\right) + P(F_i)\,\delta\left(\Lambda(u_i) - \log\frac{P_{D_i}}{P_{F_i}}\right) \tag{4}$$

An expression similar to (3) is obtained for hypothesis H_1.

The problem that is addressed is whether the implementation of the N-P test at the fusion center can decrease the probability of false alarm below the smallest individual P_F, while increasing the probability of detection beyond the largest individual P_D.

In this context, we consider three different cases.

Case I All the sensors are similar and operate at the same level of probability of false alarm and probability of detection, i.e. $P_{F_i} = P_{F_j} = P_F$ and $P_{D_i} = P_{D_j} = P_D$ for every i and j. Since all the probability distributions in (2) are the same, the N-P test leads to the following scheme at the fusion center:

With k out of N decisions favoring hypothesis H_1 the test is

$$k \overset{H_1}{\underset{H_0}{\gtrless}} t^* \tag{5}$$

where t^* is some threshold to be determined so that a certain overall false alarm probability, P_F^f, is attained at the fusion center.

The random variable k is distributed as binomial with parameters N and P_F under H_0 and as binomial with parameters N and P_D under H_1. Hence, P_F^f and the overall probability of detection P_D^f, are given by

$$P_F^f = \sum_{i=\lceil t^*\rceil}^{N} \binom{N}{i} P_F^i (1-P_F)^{N-i} \tag{6}$$

$$P_D^f = \sum_{i=\lceil t^*\rceil}^{N} \binom{N}{i} P_D^i (1-P_D)^{N-i} \tag{7}$$

where $\lceil t^* \rceil$ indicates the smallest integer exceeding t*.

Consider the following example. A system of four sensors, N=4, operating all at $P_F = 0.05$ and $P_M = 0.05$ ($P_D = 0.95$). If t* = 2, from the binomial cumulative table we get $P_F^f = 0.014$ and $P_M^f = 0.0005$ at the fusion center, i.e. a considerable improvement of the overall system.

To assess the performance of the fusion scheme further, we compare it with a centralized scheme which utilizes raw data - not decisions - from the different sensors. Let a single observation from each of the N sensors be distributed normally as

$$\begin{aligned} r_i &\sim G(0,1) \quad \text{under } H_0 \\ &\sim G(S,1) \quad \text{under } H_1 \end{aligned}$$

Assume $P_F = 0.05 = P_M$ for an individual sensor, and a $P_F^b = 0.001$ at the fusion center. This gives a $P_M^b = 0.0002$. This example shows that the best decentralized fusion scheme achieves a $(P_F^f, P_M^f) = (0.014, 0.0005)$, whereas the best centralized fusion scheme achieves a $(P_F^b, P_M^b) = (0.001, 0.0002)$ for the same sensors. Clearly the loss in power associated with transmitting highly condensed information from the sensors to the fusion center causes the degradation in the performance of the fusion scheme. As a compromise, a multibit information could be transmitted to the fusion center containing quality information related to the degree of confidence that a sensor has about its decision along with the decision itself. This situation is examined in Section II.

Table 1 gives the different N-P test thresholds that the fusion center can operate so that condition (8) is satisfied. The thresholds were found using the Interactive Fusion Algorithm (IFA) that we developed (See Appendix for a brief description of IFA). From the table it can be seen that decision fusion improves the performance of the system provided that three or more sensors are used.

For the configuration of N sensors, we are interested to know whether the N-P trest can provide with (P_D, P_F) so that

$$P_F^f \le \min_{i \in N} \{P_{F,i}\} \quad \text{and} \quad P_D^f > \max_{i \in N} \{P_{D,i}^f\} \qquad (*)$$

The next theorem shows that condition (*) can be satisfied if the randomized Neyman-Pearson test is used at the fusion center and the number of sensors N is greater than two.

Theorem 2

In a configuration of N similar sensors, all operating at the same $(P_F, P_D) = (p,q)$, the randomized N-P test at the fusion center can provide with (P_F, P_D) satisfying (*) if $N \ge 3$.

More precisely, for $N \ge 3$, the randomized N-P test can be fixed so that

$$P_F^f = P_F = p \quad \text{and} \quad P_D^f > P_D = q \qquad (**)$$

where P_F and P_D are the probability of false alarm and probability of detection at the individual sensors.

The proof of the theorem is given in the Appendix II.

Case II All the sensors operate at the same probability of false alarm level P_F, but different levels of probability of detection from each other, i.e. $P_{D_i} \neq P_{D_j}$, $i \neq j$. Without loss of generality we assume the ranking $P_{D_1} > P_{D_2} > \cdots > P_{D_N}$, from which the following ordering in the absiscae of the conditional distribution of the individual likelihood ratio results:

$$\frac{1 - P_{D_1}}{1 - P_F} < \frac{1 - P_{D_2}}{1 - P_F} < \ldots < \frac{1 - P_{D_N}}{1 - P_F} < \frac{P_{D_N}}{P_F} < \ldots < \frac{P_{D_1}}{P_F}$$

As an example, consider a five sensor system. All the sensors operate at the same level $P_F = 0.05$ but with different P_D's,

Table 2

i	1	2	3	4	5
P_{D_i}	0.95	0.94	0.93	0.92	0.91

Table 3 summarizes all the choices of thresholds of the fusion center that satisfy condition (8) as given by the IFA. A significant improvement in the system performance is achieved by fusing the individual decisions.

Case III The different sensors operate at different probabilities of false alarm and probabilities of detection , i.e. $P_{F_i} \neq P_{F_j}$ and $P_{D_i} \neq P_{D_j}$, $i \neq j$. The distribution of the cumulative likelihood ratio of the fusion center is obtained numerically as in Case II, and the threshold t* is found to satisfy a given P_F^f. Ideally, the threshold t* must be chosen so that

$$P_F^f < \min_i \{P_{F_i}\} \text{ and } P_D^f > \max_i \{P_{D_i}\} \tag{8}$$

The following examples illustrate the procedure.

We consider three different systems with five, four and three sensors. Each system results by eliminating the sensor with the lowest P_D from the system that has one more sensor. For the five sensor system the sensors (P_D,P_F)'s are given in Table 4.

Table 4

i	1	2	3	4	5
P_{D_i}	0.95	0.94	0.93	0.92	0.91
P_{F_i}	0.05	0.04	0.03	0.02	0.01

Table 5 summarizes the results as obtained by IFA. In all cases, a significant improvement in the performance of the system is achieved from fusing the decisions.

SECTION II

Transmission of Decisions Along with Quality Information

Consider the case where the j-th sensor transmits quality information bits to the fusion center about its decision along with the decision itself. The simplest case corresponds to the transmission of binary, {0,1}, quality information indicating the degree of confidence that the sensor has on the decision that it transmits. Under this scenario, a bit one will indicate "confidence", whereas a bit zero "no confidence". Figure 1 help illustrate how the binary quality bit "c" is defined. A strip (T_L, T_u) about the threshold T of an individual sensor is designated as region of "no confidence" and the bit c = 0 is transmitted along with the decision when the observation r falls into this region. The complimentary to (T_L, T_u) regions are considered "confidence" regions and the bit c = 1 is transmitted along with the decision when the observations fall into one of the two regions.

The joint probability distribution at (u,c) (skipping the sensor index for simplicity) can be easily obtained from: (refer Fig. 1)

$$P(u,c|H_k) = P(c|u,H_k)\, P(u|H_k) \quad , \quad k = 0,1 \tag{9}$$

where $P(u|H_k)$, $u = \pm 1$ and $k = 0,1$ is specified by P_F and P_D.

Hence, for every sensor

$$P(u=i,\ c=j|H_k) = c_{ji}^{k}\ P\ (u=i|H_k) \tag{10}$$

and

$$\Lambda(u,c) = \frac{P(u,c|H_1)}{P(u,c|H_0)} = \frac{c^1_{ji}\ P(u=i|H_1)}{c^0_{ji}\ P(u=i|H_0)} \tag{11}$$

from which $P(\Lambda(u,c)|H_1)$ and $P(\Lambda(u,c)|H_0)$ can be obtained.

Therefore, the probability distribution of the LR at the fusion center is given by the convolution

$$P(\log \Lambda(u,c)|H_k) = P(\log \Lambda(u_1,c_1)|H_k) * \ldots * P(\log \Lambda(u_N,c_N)|H_k) \tag{12}$$

In the case that all the sensors operate at the same level (P_F, P_D) the mathematics simplify somewhat, leading to

$$P^f_F = \sum_{k=t^*_1}^{N} \sum_{n=t^*_2}^{k} \sum_{m=t^*_3}^{N-k} [\binom{k}{n}[c^0_{11}]^n [1-c^0_{11}]^{k-n} \binom{N-k}{m}[c^0_{10}]^m$$

$$\cdot[1-c^0_{10}]^{N-k-m} \binom{N}{k} P^k_F(1-P_F)^{N-k}] \tag{13}$$

P^f_D is obtained by a similar to (13) expression with P_D in place of P_F and the index 1 instead of 0 above c_{ij}. The thresholds t^*_1, t^*_2 and t^*_3 are to be determined to satisfy a given probability of false alarm at the fusion center.

From (13) it is seen that a superior performance in regards with (P^f_F, P^f_D) can be achieved when quality information is transmitted along with the decisions. To show that, consider the example of Case I in Section I. The threshold at each sensor is set to $t_i = 1.64$ to satisfy P_F. Using Figure 1 and the previous equations, we obtain that for $t_{L,i} = 0.8t_i = 1.312$ and $t_{u,i} = 1.2\ t_i = 1.968$, the c^k_{ij}'s are given in Table 6.

Table 6

C_{ij} \ H_k	1	0
11	0.948	0.46
01	0.052	0.54
00	0.52	0.047
10	0.48	0.953

Using the Interactive Fusion Algorithm (IFA), it follows that there is a choice of thirty-three (33) different thresholds that the fusion center can operate so that condition (8) is satisfied, Table 7. For a comparable $P^f_M = 0.0005$, the $P^f_F = 0.0048$ when quality bit information is transmitted as opposed to $P^f_F = 0.014$ without quality information.

It is interesting to notice that, fusion of the decisions improves the performance of the overall system even in the case of two sensors when quality information bits are transmitted along with the decisions, Table 8.

CONCLUSIONS

The problem of fusing decisions from N independent sensors in a fusion center was considered. We assumed that each sensor transmits its decision to the fusion center. The decision of each individual sensor is based on the Neyman-Pearson (N-P) test. The fusion center formulates the likelihood ratio (LR) using all the received decisions and decides on which hypothesis is true using the N-P test also. A number of examples are given to demonstrate the significant improvement in the performance of the system from decision fusing.

Moreover, the problem of decision fusion when, along with their decisions, the sensors transmit quality information bits indicating their confidence on their decisions was considered. The solution of the problem in the N-P sense was provided. Several numerical examples demonstrated that use of quality information bits can improve the performance of the center considerably. An Interactive Fusion Algorithm (IFA) was developed to solve the fusion problem numerically.

REFERENCES

1] Tenney, R. R. and Sandell, N. R., Jr., "Detection with Distributed Sensors," IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-17, July 1981, pp. 501-510.

2] Sanjadi, F. A., "Hypotheses Testing in A Distributed Environment," IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-22, No. 2, March 1986, pp. 134-137.

3] Teneketzis, D. and Varaiya, P., "The Decentralized Quickest Detection Problem," IEEE Transactions on Automatic Control, Vol. AC-29, No. 7, July 1984, pp. 641-644.

4] Teneketzis, D., "The Decentralized Wald Problem," IEEE 1982 International Large-Scale Systems Symposium, Virginia Beach, October 1982, pp. 423-430.

5] Tsitsiklis, J. and Athans, M., "On the Complexity of Distributed Decision Problems," IEEE Transactions on Automatic Control, Vol. AC-30, No. 5, May 1985, pp. 440-446.

6] Chair, Z. and Varshney, P. K., "Optimal Data Fusion in Multiple Sensor Detection Systems," IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-22, No. 1, January 1986, pp. 98-101.

APPENDIX I

The Interacting Fusion Algorithm (IFA) receives as data the number of sensors, their (P_F,P_D) levels and the c_{ij}^k quality information parameters if the sensors transmit quality information bits along with their decisions. It then computes the LR pdf at the fusion center conditioned on each hypothesis. After it computes the pdf, it asks the user which option he/she prefers. The alternative options are:

a) Display of the entire pdf.

b) Threshold computation for a given P_F^f and display of the corresponding P_D^f.

c) Determination of the thresholds that satisfy condition (16).

d) Threshold computation for a given P_D^f and display of the corresponding P_F^f.

e) Elimination of one or more sensors and repetition of the algorithm.

f) Quit.

APPENDIX II

For the configuration of N sensors, we are interested to know whether the N-P test can provide with (P_D,P_F) so that

$$P_F^f \le \min_{i\varepsilon N} \{P_{F,i}\} \quad \text{and} \quad P_D^f > \max_{i\varepsilon N} \{P_{D,i}^f\} \qquad (*)$$

Theorem 2

In a configuration of N similar sensors, all operating at the same $(P_F,P_D) = (p,q)$, the randomized N-P test at the fusion center can provide with (P_F,P_D) satisfying (*) if $N \ge 3$.

More precisely, for $N \ge 3$, the randomized N-P test can be fixed so that

$$P_F^f = P_F = p \quad \text{and} \quad P_D^f > P_D = q \qquad (**)$$

where P_F and P_D are the probability of false alarm and probability of detection at the individual sensors.

Proof First we show that for N = 2, condition (*) cannot be satisfied with the second inequality as a strict one. Then we prove that for N = 3, the randomized N-P test satisfies condition (*). By using the fact that the probability of detection is maximized by the N-P test for all mappings from the observation space into decision space and for fixed probability of false alarm, we prove by induction that condition (*) is satisfied for all $N \geq 3$.

Let $N = 2$ and $(P_F, P_D) = (p,q)$ for both sensors. According to Eqs. (4), (5), (6) and (9), the likelihood ratio distribution at the fusion center under hypothesis H_0 and H_1 are plotted in Fig. 2.

From Fig. 2 we see that

$$p^2 < p < 2p(1-p) + p^2 \tag{1}$$

Thus, in order to satisfy $P_F = p$, the randomized N-P test is defined by

$$p^2 + \omega 2p(1-p) = p \tag{2}$$

where $0 < \omega < 1$. Solving (2) we obtain $\omega = 0.5$, independent of p. Since P^f_D is determined by a symmetric expression to (2), [see Fig. 3], $P^f_D = q$ for $\omega = 0.5$. Hence, condition (**) cannot be satisfied for $N = 2$.

Let $N = 3$. The distribution of the likelihood ratio under H_0 and H_1 are given in Figure 3. From Fig. 3 we see that

$$p^3 + 3p^2(1-p) < p \tag{3}$$

for $0 < p < 0.5$. The r.h.s. of inequality (3) is greater than p for $p > 0.5$. Hence, since $P_F < 0.5$, the randomized N-P test at the fusion center is determined by

$$p^3 + 3p^2(1-p) + \omega 3p(1-p)^2 = p \tag{4}$$

from which

$$\omega = \frac{1}{3} - \frac{p}{3(1-p)} \tag{5}$$

which is a positive fraction for $0 < p < 0.5$.

Since P^f_F at the fusion center is given by a similar to (4) expression with q in place of p, and $q > 0.5$, it follows from (3) that $P^f_F > q$, which proves the theorem for $N = 3$.

Assume that for that the ramdomized N-P test satisfies condition (**) for an arbitrary number of sensors, N. We will show that it will also satisfy the condition for N + 1, and thus complete the induction and the proof of the theorem.

Let $D_N = \{D^1, D^2, \ldots, D^N\}$ designate the set of the decisions from the N sensors that are available to the fusion center at the time of the decision. All the sensors operate at the same level (p,q) satisfying condition (**). Let $f_N(D_N)$ designate some decision rule at the fusion center operating at fixed probability of false alarm p. Let $f^{N-P}_N(D_N)$ designate the randomized N-P test at the fusion center at level p. For fixed probability of false alarm, the probability of detection at the fusion center (power of test) is maximized for the N-P decision rule among all possible decision rules.

Let $D_{N+1} = \{D_N, D^{N+1}\}$ designate the decision ensemble of N + 1 similar sensors all operating at the same level (p,q). Then by choosing $f_{N+1}(D_{N+1}) = f^{N-P}_N(D_N)$,

$$P_D(f^{N-P}_{N+1}(D_{N+1}) = \max_{f_{N+1}(D_{N+1})} P_D(f_{N+1})) \geq P_D(f^{N-P}_N(D_N)) \tag{6}$$

from which it follows that

$$P^f_{D,\,N+1} \geq P^f_{D,N} > \tag{7}$$

TABLE 1

Decision Fusion : 5 Sensor System
Sensors PF : Equal x Unequal ___
Sensors PD : Equal x Unequal ___

Threshold @ Fusion Center	Probability of False Alarm @ Fusion Center	Probability of Detection @ Fusion Center
PDMAX = .95000		PFMIN = .50000E-01
t*	PD	PF
6859.0	.977407	.300000E-04
19.000	.998842	.115812E-02
.52631E-01	.999970	.225925E-01
	1 SENSOR OFF	
PDMAX = .95000		PFMIN = .50000E-01
t*	PD	PF
361.00	.985981	.481250E-03
1.0000	.999519	.140187E-01
	2 SENSORS OFF	
PDMAX = .95000		PFMIN = .50000E-01
t*	PD	PF
19.000	.992750	.725000E-02

TABLE 3

Decision Fusion : 5 Sensor System
Sensors PF : Equal x Unequal ___
Sensors PD : Equal ___ Unequal x

Threshold @ Fusion Center	Probability of False Alarm @ Fusion Center	Probability of Detection @ Fusion Center
PDMAX = .95000		PFMIN = .50000E-01
t*	PD	PF
6163.2	.957817	.300000E-04
53.004	.963797	.142812E-03
45.880	.968973	.255625E-03
40.339	.973523	.368437E-03
38.907	.977913	.481250E-03
34.208	.981772	.594062E-03
32.081	.985391	.706874E-03
29.610	.988731	.819687E-03
28.207	.991913	.932499E-03
24.416	.994668	.104531E-02
20.705	.997003	.115812E-02
.20998	.997454	.330156E-02
.17806	.997835	.544500E-02
.15413	.998165	.758843E-02
.14683	.998480	.973187E-02
.13552	.998771	.118753E-01
.12709	.999043	.140187E-01
.11174	.999282	.161622E-01
.10778	.999513	.183056E-01
.94760E-01	.999717	.204490E-01
.82023E-01	.999892	.225925E-01

TABLE 5

Decision Fusion : 5 Sensor System
Sensors PF : Equal ___ Unequal x
Sensors PD : Equal ___ Unequal x

Threshold @ Fusion Center	Probability of False Alarm @ Fusion Center	Probability of Detection @ Fusion Center
PDMAX = .95000		PFMIN = .10000E-01
t*	PD	PF
57882.	.957817	.269200E-05
426.86	.960153	.816400E-05
373.63	.962908	.155360E-04
358.72	.966248	.248480E-04
284.83	.969430	.360200E-04
273.46	.973289	.501320E-04
239.36	.977840	.691439E-04
160.34	.981459	.917159E-04
153.94	.985848	.120228E-03
134.74	.991024	.158640E-03
102.72	.997003	.216852E-03
.99369	.997179	.393780E-03
.75752	.997382	.661908E-03
.66305	.997622	.102314E-02
.63660	.997912	.147942E-02
.42643	.998143	.202115E-02
.37325	.998416	.275098E-02
.35836	.998746	.367287E-02
.28454	.999061	.477889E-02
.27319	.999442	.617598E-02
.23912	.999892	.805816E-02
	1 SENSOR OFF	
PDMAX = .95000		PFMIN = .20000E-01
t*	PD	PF
1129.9	.976981	.150400E-03
4.6908	.979548	.697600E-03
4.1058	.982575	.143480E-02
3.9420	.986246	.236600E-02
3.1300	.989742	.348320E-02
3.0051	.993983	.489440E-02
2.6303	.998984	.679560E-02
	2 SENSORS OFF	
PDMAX = .95000		PFMIN = .30000E-01
t*	PD	PF
32.222	.989720	.458000E-02

TABLE 7

Decision Fusion : 4   Sensor System with Quality Bits
Sensors PF : Equal x   Unequal ___
Sensors PD : Equal x   Unequal ___

Threshold @ Fusion Center	Probability of False Alarm @ Fusion Center	Probability of Detection @ Fusion Center
PDMAX = .95000		PFMIN = .50000E-01
t*	PD	PF
62318.	.956002	.175551E-05.
20357.	.960940	.199808E-05
9390.7	.961918	.210220E-05
2988.7	.963462	.261876E-05
2911.9	.980782	.856706E-05
951.21	.981595	.942131E-05
926.74	.990711	.192580E-04
438.79	.990738	.193191E-04
302.74	.990880	.197900E-04
139.65	.990937	.201943E-04
136.06	.992362	.306685E-04
44.446	.992406	.316713E-04
43.303	.993906	.663133E-04
42.189	.998114	.166041E-03
20.503	.998114	.166055E-03
14.146	.998129	.167161E-03
13.782	.998524	.195805E-03
6.5253	.998525	.195924E-03
6.3575	.998577	.204121E-03
4.5021	.998579	.204578E-03
2.0768	.998580	.204970E-03
2.0234	.998662	.245637E-03
1.9713	.999354	.596850E-03
.66097	.999355	.597499E-03
.64397	.999398	.664750E-03
.62741	.999762	.124555E-02
.29706	.999763	.124796E-02
.21036	.999763	.124850E-02
.20495	.999771	.128557E-02
.94544E-01	.999772	.130148E-02
.92113E-01	.999810	.171378E-02
.66951E-01	.999810	.171395E-02
.30090E-01	.999811	.175343E-02
.29316E-01	.999851	.311705E-02

TABLE 8

Decision Fusion : 2   Sensor System with Quality Bits
Sensors PF : Equal x   Unequal ___
Sensors PD : Equal x   Unequal ___

Threshold @ Fusion Center	Probability of False Alarm @ Fusion Center	Probability of Detection @ Fusion Center
PDMAX = .95000		PFMIN = .50000E-01
t*	PD	PF
1.0654	.951900	.696499E-02
1.0380	.995129	.486111E-01

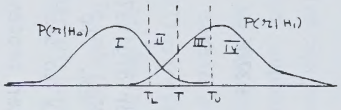


FIGURE 1

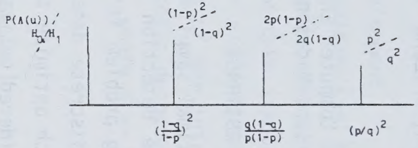


FIGURE 2

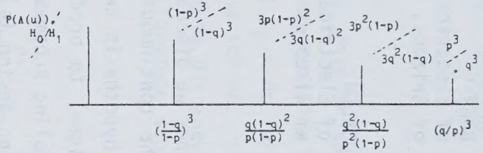


FIGURE 3

DESIGN OF OPTIMAL SAMPLING SCHEMES FOR THE GAUSSIAN HYPOTHESIS TESTING PROBLEM

RANDALL K. BAHR
Department of Electrical and Computer Engineering,
University of Arizona, Tucson, Arizona 85721

JAMES A. BUCKLEW
Department of Electrical and Computer Engineering,
University of Wisconsin-Madison, Madison, Wisconsin 53706

ABSTRACT

In this paper we discuss optimal sampling schemes for digital implementation of the continuous time detection theory problem. To digitally implement this hypothesis testing problem it is necessary to sample a continuous time signal to produce discrete data for computer processing. We investigate sampling schemes which optimize the digital detector performance. A sampling design is considered optimal if it minimizes the asymptotic probability of error.

I. Introduction

With the advent of microprocessor based analysis systems it has become highly desirable to translate continuous time communication problems into analogous discrete time problems via data sampling. Consider the classical continuous time detection theory problem:

$$\begin{aligned} H_1&: x(t) = s(t) + n(t) \\ H_0&: x(t) = \phantom{s(t) + {}} n(t) \end{aligned} \tag{1.1}$$

where $s(\cdot)$ and $n(\cdot)$ are independent wide sense stationary zero mean Gaussian random processes. The solution consists in implementing a likelihood ratio (LR) test involving the filtering and cross-correlation of the received signal $x(\cdot)$. This requires continuous time integration techniques. Suppose this approach is inconvenient or impossible and we must build a digital receiver to decide between the two hypotheses. We can easily sample the signal at a much faster rate than our processor can implement the LR test; therefore, it is necessary to judiciously sample the signal in order to maintain detector integrity.

Our study is closely allied with [1]. Here the authors investigate sampling designs for detecting a sure signal in Gaussian noise. Both deterministic and random sampling schemes are considered.

In our paper we investigate sampling schemes for the random signal in random noise problem. The criterion for optimal detector performance is the asymptotic probability of error.

II. Problem Statement and Development

Let us suppose our processor can handle only N samples every τ seconds. We constrain our sampling designs to be periodic in τ, i.e., if we take a sample at t_1 then we take samples at $t_1 + n\tau$ for $n = 1,2,3,\ldots$ Figure 1 illustrates the N per τ second sampling scheme. Let Π designate the sampling design consisting of the intra-sample distances $\{a_k\}_{k=1}$ with $a_k > 0$. To analyze the asymptotic detector performance suppose we have a large number of samples, say K, where K >> N and K is an integer multiple of N. This represents data based on the restriction of only N samples every τ seconds but consisting of a very long data record.

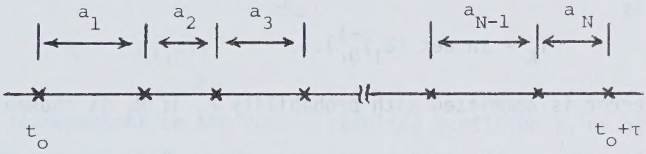


Figure 1

Let π_0 and π_1 be the a priori probabilities of H_0 and H_1 respectively. It is well-known that the LR test [4] is

$$\overline{x}'(\Sigma_0^{-1} - \Sigma_1^{-1})\overline{x} \geq T \tag{2.1a}$$

$$T = \ln \det(\Sigma_1\Sigma_0^{-1}) + \ln \frac{\pi_0}{\pi_1} \tag{2.1b}$$

where Σ_i is the covariance matrix of the samples under hypothesis H_i, $\overline{x}$ is the column vector of samples, and the prime denotes transposition. By a transformation of coordinates $\overline{x}$ may be whitened, effecting a simultaneous diagonalization of the two covariance matrices. The test statistic becomes

$$Z_K = \sum_{p=1}^{K} (1 - \frac{1}{\lambda_p}) y_p^2 \gtrless T \tag{2.2}$$

where

under H_0: y_p are iid $N(0,1)$

under H_1: y_p are independent $N(0,\lambda_p)$

and $\{\lambda_p\}_{p=1}^{K}$ are the eigenvalues of $\Sigma_1\Sigma_0^{-1}$ [4,p. 33]. Therefore (2.1) represents the digital formulation of the continuous time detection problem (1.1) under the sampling scheme $\{a_k\}_{k=1}^{N}$. For purposes of analyzing the detector's asymptotic behavior the samples are whitened and we arrive at the new formulation (2.2). The asymptotic behavior of the detector reduces to a study of the asymptotic behavior of the eigenvalues of $\Sigma_1\Sigma_0^{-1}$.

The criterion of optimal detector design is the probability of error. Under equally likely hypotheses, the threshold minimizing the probability of error is

$$T_K = \ln \det (\Sigma_1\Sigma_0^{-1}). \tag{2.3}$$

A type 1 error is committed with probability P_1^K if H_1 is chosen when H_0 is true

$$P_1^K = P\{\text{say } H_1 \text{ true } | H_0 \text{ is true}\}. \tag{2.4}$$

Computation of P_1^K is equivalent to finding the probability that the LR under H_0 exceeds the threshold T.

Under quite general conditions, it is known that P_1^K decreases exponentially with increasing sample size,

$$P_1^K = C \exp(-K(M-\varepsilon)) \tag{2.5}$$

for some positive constant C and arbitrary positive ε. M represents the exponential rate of decrease and has traditionally been computed via normal approximations as a Chernoff bound. Clearly the best asymptotic performance is obtained when M is maximized. To analyze P_1^K asymptotically we turn to the whitened test statistic (2.2) and observe that

$$P_1^K = P\{Z_K \geq T_K\}. \tag{2.6}$$

The asymptotic behavior of (2.6) is handled quite nicely by a branch of probability known as large deviation theory. Briefly, it can be shown that [3]

$$\overline{\lim}\ \frac{1}{K}\log P_1^K = \overline{\lim}\ \frac{1}{K}\log P\{\frac{1}{K} Z \geq \frac{T_K}{K}\} \leq - I([T,\infty)) \tag{2.7}$$

(this is simply (2.5) in another guise) where

$$T = \lim \frac{T_K}{K} \tag{2.8}$$

$$I([T,\infty)) = \inf\{I(z): z\varepsilon[T,\infty\ )\} \tag{2.9a}$$

$$I(z) = \sup_{\Theta\varepsilon R} \{\Theta z - c(\Theta;\Pi)\}. \tag{2.9b}$$

$I(\cdot)$ is known as the rate function for $\{Z_K\}$, as it sets the exponential rate of tail probability decrease, and is connected to the asymptotic distribution of $\Sigma_1\Sigma_0^{-1}$ via the Legendre/Fenchel transform (2.9b) of the moment generating function (MGF) $c(\Theta;\Pi)$,

$$c(\Theta;\Pi) = \lim_{K\to\infty} \frac{1}{K}\log E(\exp(\Theta Z_K)). \tag{2.10}$$

E is the expectation operator and Z_K are defined by (2.2). Noting that

$$M = I([T,\infty)) \tag{2.11}$$

and the MGF is dependent on the chosen sampling partition Π, we may investigate the dependence of M on the sampling scheme via (2.10), (2.9a), and (2.9b).

In the next section it will be shown that $c(\Theta;\Pi)$ may be written in terms of the asymptotic distribution of the eigenvalues of $\Sigma_1\Sigma_0^{-1}$. Designate the K eigenvalues of $\Sigma_1\Sigma_0^{-1}$ by λ_p^K, p = 1,2,...K and we say that the $\{\lambda_p^K\}$ are asymptotically equally distributed (aed) with respect to a function f if

$$\lim_{K\to\infty} \frac{1}{K}\sum_{p=1}^{K} F(\lambda_p^K) = \int_0^1 F(f(x))dx. \tag{2.12}$$

F is any continuous function defined on the interval $[\delta_1,\delta_2]$ where δ_1 and δ_2 are the essential lower and upper bounds of f. The MGF is written as a limit of a sum of a function of the eigenvalues and thus we need to find the f which asymptotically distributes the eigenvalues of $\Sigma_1\Sigma_0^{-1}$. To do this, we use the standard technique of circulant approximation of Toeplitz matrices. The more readily analyzed circulant matrices yield simple expressions for the distribution function f. The proof of our main theorem relies on the treatment of asymptotic equal distribution of eigenvalues found in Gray [5] and the more mathematical Grenander and Szego [6].

In the next section we present the solution to the sampled problem. We find the sampling scheme Π which maximizes the asymptotic exponential rate M under two assumptions. First, assume that the signal $s(\cdot)$ is a zero mean second order wide sense stationary Gaussian random process with continuous spectral density $S(\cdot)$ and autocorrelation function $r(\cdot)$

$$r(\tau) = \int_{-\infty}^{\infty} S(x)e^{-2\pi i x\tau}\, dx. \tag{2.13}$$

Further, assume that the noise in the sampled problem is white, i.e., is uncorrelated from sample to sample regardless of the sampling scheme and each sample has fixed variance σ_0^2. Though this situation can not arise from sampling a second order random process, a whiteness assumption is a widely used engineering approximation and will be adhered to in this paper. Second, assume that the asymptotic quantity of optimization is the probability of type 1 error. In other words, choose the asymptotic threshold T to minimize the probability of error for a particular sample design Π, then adjust Π to maximize the rate function $I(\cdot)$ for the type 1 error.

III Solution

To find an expression for the MGF, which in turn determines the rate function $I(\cdot)$, it is first necessary to investigate the asymptotic distribution of the eigenvalues of $\Sigma_1\Sigma_0^{-1}$. Under the whiteness assumption of the sampled noise, Σ_0^{-1} is diagonal with entries $\frac{1}{\sigma_0^2}$. Further, the correlation matrix under H_1, Σ_1, can be written as a sum of a diagonal matrix with entries σ_0^2 and a matrix Σ which consists of sampled values of the signal autocorrelation function $r(\cdot)$. Thus if the eigenvalues of Σ are $\{\lambda_p\}$ the eigenvalues of $\Sigma_1\Sigma_0^{-1}$ are

$$\frac{\sigma_0^2 + \lambda_p}{\sigma_0^2}. \tag{3.1}$$

Hence we are reduced to studying the properties of Σ alone. Σ while positive definite is not Toeplitz since the intra-sample distances, a_k, are not necessarily equal; however, because of the periodicity of the sampling scheme, Σ does admit a "modified" Toeplitz structure. Designate the first row of Σ as 1 then the 1 + kN row is the first row shifted by kN for $k = 1,2,.. \frac{K}{N} - 1$. Similarly the 2 + kN row is the 2^{th} row ($n \leq N$) shifted by kN. This matrix may be brought into a manageable form by rearrangement of the rows and columns via a similarity transform. This similarity transform does not alter the eigenvalues of the matrix but will produce a matrix which consists of blocks of Toeplitz matrices,

$$\Sigma \text{ is similar to } \begin{bmatrix} A_{11} & A_{12} & \cdots\cdots & A_{1N} \\ A_{12} & A_{11} & A_{23} \cdots & A_{2N} \\ A_{13} & A_{13} & A_{11} \cdots & A_{3N} \\ \vdots & & & \vdots \\ A_{1N} & \cdots\cdots & \cdots\cdots & A_{11} \end{bmatrix} = \hat{\Sigma} \tag{3.2}$$

where each A_{pq} is a $\frac{K}{N} \times \frac{K}{N}$ non-symmetric Toeplitz matrix consisting of sampled values of the signal correlation function $r(\cdot)$. This particular structure allows us to analyze each Toeplitz block separately by a circulant approximation. The individual asymptotic distributions of each block are then used to produce the asymptotic distribution of eigenvalues for the entire matrix $\hat{\Sigma}$ (and hence Σ). We have the following theorem.

Theorem: The eigenvalues of $\hat{\Sigma}$ are asymptotically equally distributed according to

$$\lim_{K\to\infty} \frac{1}{K} \sum_{p=1}^{K} F(\lambda_p) = \frac{1}{N} \int_0^1 \sum_{p=1}^{N} F(\lambda_p(x))\,dx \tag{3.3}$$

for continuous functions $F(\cdot)$. The $\{\lambda_p(x)\}$ for fixed $x \in [0,1]$ are the N solutions to the eigenvalue problem

$$\begin{bmatrix} f_{11}(x) & f_{12}(x) & \cdots\cdots & f_{1N}(x) \\ f_{12}^*(x) & f_{11}(x) & \cdots\cdots & f_{2N}(x) \\ \vdots & & & \vdots \\ f_{1N}^*(x) & f_{2N}^*(x) & \cdots\cdots & f_{11}(x) \end{bmatrix} \bar{y} = \lambda_p(x)\,\bar{y}\,. \tag{3.4}$$

The $\{\lambda_p(\cdot)\}$ are real continuous functions. The functions f_{pq} are given by

$$f_{pq}(x) = S_\Delta^{\,F}(x) \tag{3.5a}$$

where

$$S_\Delta^F(x) = \frac{1}{\tau} \sum_{k \in Z} S_\Delta\left(\frac{x-k}{\tau}\right) \quad 0 \leq x \leq 1 \tag{3.5b}$$

and

$$S_\Delta(x) = S(x)\, e^{2\pi i x\Delta} . \tag{3.5c}$$

$S(\cdot)$ is the spectral density of the signal $s(\cdot)$. The value of Δ may be obtained from Table 1.

Table 1

p,q	Δ	
1,1	0	
k,k+1	a_k	k=1,..,N-1
k,k+2	$a_k + a_{k+1}$	k=1,..,N-2
k,k+3	$a_k + a_{k+1} + a_{k+2}$	k=1,..,N-3
.		
k,k+ℓ	$a_k + a_{k+1} + \ldots + a_{k+\ell}$	k=1,..,N-ℓ
.		
1,N	$a_1 + a_2 + \ldots + a_{N-1}$	

From this theorem and (2.14) it is immediate that the asymptotic eigenvalue distribution of $\Sigma_1 \Sigma_0^{-1}$ is given by

$$\frac{1}{N} \int_0^1 \sum_{p=1}^{N} F\left(\frac{\sigma_0^2 + \lambda_p(x)}{\sigma_0^2}\right) dx . \tag{3.6}$$

Thus the asymptotic behavior of $\Sigma_1 \Sigma_0^{-1}$ is determined by the folded spectra S_Δ^F and ultimately by the partition Π. It now remains to relate the MGF $c(\Theta,\Pi)$ to the $\{\lambda_p(x)\}_{p=1}^N$. Under H_0, Z_K consists of a weighted sum of iid $N(0,1)$ random variables, so the MGF (2.10) becomes

$$\begin{aligned}
c(\Theta;\Pi) &= \lim_{K\to\infty} \frac{1}{K} \log E\left[\exp \Theta \sum_{p=1}^{K} \left(1 - \frac{1}{\lambda_p}\right) y_p^2\right] \\
&= \lim_{K\to\infty} \frac{1}{K} \sum_{p=1}^{K} \left[-\frac{1}{2} \log\left(1 - 2\Theta\left(1 - \frac{1}{\lambda_p}\right)\right)\right] \\
&= -\frac{1}{2N} \int_0^1 \sum_{p=1}^{N} \log\left(1 - 2\Theta\left(\frac{\lambda_p(x)}{\sigma_0^2 + \lambda_p(x)}\right)\right) dx \\
&= -\frac{1}{2N} \int_0^1 \log \prod_{p=1}^{N} \left(1 - 2\Theta\left(\frac{\lambda_p(x)}{\sigma_0^2 + \lambda_p(x)}\right)\right) dx .
\end{aligned} \tag{3.7}$$

By an application of Ellis' Theorem II.2 [3,p.3], the asymptotic rate for the probability of type 1 error may be found

$$\begin{aligned}\overline{\lim_{K\to\infty}}\ \frac{1}{K}\log P_1^K &= \overline{\lim_{K\to\infty}}\ \frac{1}{K}\log P\{Z_K \ge T_K\}\\ &= \overline{\lim_{K\to\infty}}\ \frac{1}{K}\log P\{\tfrac{1}{K}Z_K \ge \tfrac{T_K}{K}\}\\ &\le -I([T,\infty)) = -I(T)\ . \end{aligned} \tag{3.8}$$

Equality in the last line follows from the convexity of the rate function. Combining (3.7) and (2.9b) yields the rate function

$$I(T) = \sup_{\Theta\epsilon R}\ \{\Theta T + \frac{1}{2N}\int_0^1 \log \prod_{p=1}^{N} (1-2\Theta(\frac{\lambda_p(x)}{\sigma_0^2 + \lambda_p(x)}))\ dx\}. \tag{3.9}$$

To perform the above supremum, take the derivative with respect to θ and set equal to zero to obtain

$$T = \frac{1}{N}\sum_{p=1}^{N}\int_0^1 \frac{\lambda_p(x)}{\sigma_0^2 + (1-2\Theta)\lambda_p(x)}\ dx\ . \tag{3.10}$$

Solve this equation for Θ_T then substitute Θ_T into

$$I(T) = \Theta T + \frac{1}{2N}\int_0^1 \log \prod_{p=1}^{N} (1-2\Theta(\frac{\lambda_p(x)}{\sigma_0^2 + \lambda_p(x)}))\ dx \tag{3.11}$$

to give the correct result. Each partition Π gives an asymptotic threshold T and a rate I(T) via (2.8) and (3.9). T is found by considering

$$\begin{aligned} T &= \lim_{K\to\infty} \frac{T_K}{K}\\ &= \lim_{K\to\infty} \frac{1}{K}\ln\det \Sigma_1\Sigma_0^{-1}\\ &= \frac{1}{N}\int_0^1 \log \prod_{p=1}^{N} (1+\frac{\lambda_p(x)}{\sigma_0^2})\ dx. \end{aligned} \tag{3.12}$$

We may now state the following algorithm for determining the optimal sampling scheme Π.

1. Select positive numbers $a_1,\ldots,a_N$ such that

$$\sum_{p=1}^{N} a_p = \tau. \tag{3.13}$$

2. Compute the folded spectra S_Δ^F referring to Table 1 for the values of Δ.
3. Compute the asymptotic eigenvalue distribution functions $\{\lambda_p(x)\}$.
4. Find $I(T)$ via (3.9) and (3.10).
5. Modify the partition Π to maximize $I(T)$.

IV. Conclusions

In this paper we have studied the design of optimal sampling schemes to implement the continuous time detection problem by a digital processor. Because the signal $x(\cdot)$ may be sampled at a much faster rate than the processor can implement a LR test it is necessary to develop sampling schemes which maintain the integrity of the detection process. With T selected to minimize the probability of error, we state a specific algorithm to minimize the probability of type 1 error as a function of sample partition.

Clearly there exists no closed form solution for the partition values $a_1,\ldots,a_N$. These must be found by computer implementation of the given algorithm. Note that specifically for N less than five there exist simple closed form expressions for $\{\lambda_p(x)\}$ [2, p. 49] but for higher order N the eigenvalues would have to be computed via a standard matrix routine. The functions $\lambda_p(\cdot)$ need only be computed on demand for specific x as the various integrals would be computed by, say, a Romberg integration technique.

The circulant approximation techniques employed to prove our theorem do not easily generalize the asymptotic results of this paper to the case of non-white additive noise. In contrast to the case of circulant matrices, which may all be diagonalized by a single matrix of complex exponentials, the circulant approximation to $\hat{\Sigma}$ can not be diagonalized by a fixed matrix. It appears that the non-white noise problem is considerably more difficult.

References

[1] S. Cambanis and E. Masry, "Sampling designs for the detection of signals in noise," IEEE Transactions on Information Theory, Vol. IT-29, No. 1, pp. 83-104, January 1983.

[2] L. E. Dickson, A First Course in The Theory of Equations. New York: John Wiley and Sons, Inc., 1922.

[3] R. S. Ellis, "Large deviations for a general class of random vectors," The Annals of Probability, Vol. 12, No. 1, pp. 1-12, 1984.

[4] K. Fukunaga, Introduction to Statistical Pattern Recognition. New York: Academic Press, 1972.

[5] R. M. Gray, "On the asymptotic eigenvalue distribution of Toeplitz matrices," IEEE Transactions on Information Theory, Vol. IT-18, No. 6, pp. 725-730, November 1972.

[6] U. Grenander and G. Szego, Toeplitz Forms and Their Applications. New York: Chelsea Publishing Company, 1958.

DESIGN ALGORITHM FOR VARIABLE-LENGTH VECTOR QUANTIZERS.

Jean-Pierre ADOUL and Marc DELPRAT
CRCS, University of Sherbrooke
Sherbrooke, P.Q. CANADA

ABSTRACT

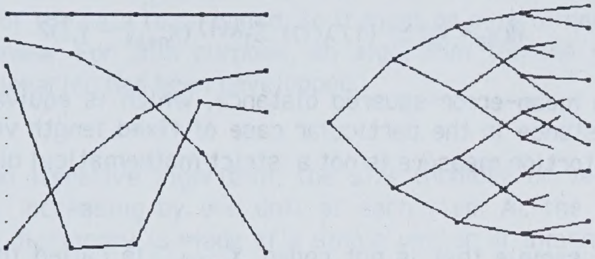
The paper introduces the concept of Variable-Length Vector Quantization (VLVQ) as an efficient source coding technique. The basic principle of VLVQ is to use variable dimension vectors to quantize variable length blocks. For the purpose of studying VLVQ a formal frame work is introduced, which enables comparison between Vector Quantization (VQ) of different lengths. In particular, it is shown that this formalism cover the vector quantization. An algorithm for designing VLVQ dictionaries from a training set is proposed. Its mechanism is detailed and its performances are carefully studied for different signals and different working conditions.

VARIABLE-LENGTH VECTOR QUANTIZATION

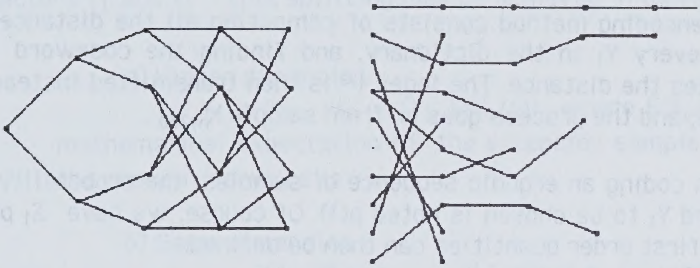
Consider the problem of quantizing a semi infinite sequence of real numbers (e.g.: samples) from some source $\{x_n \mid n = 0, 1, 2, ...\}$. Scalar Quantization (SQ) consists of quantizing one real number at a time. Vector Quantization (VQ) on the other hand quantizes M real numbers simultaneously and therefore proceeds by blocks. VQ, formerly called block quantization, has had a renewed interest with the use of the so-called K-means algorithm for VQ-design from a training set [1,2]. In VLVQ the quantizer decides how many real numbers should be quantized simultaneously for a particular situation.

VLVQ exhibits one extra degree of freedom with respect to SQ or VQ, namely the dimension of the vector or, equivalently, the block length. Suppose the sequence of real numbers represents some physical signal (e.g.: speech, video...) the extra degree of freedom allows the quantizer to "take more time" (i.e.: shorter block lengths) in quantizing transitions that are rare and unforeseeable while "catching up" (i.e.: longer block lengths) during stationary segments.

In the terminology of Jayant and Noll [3, pp 428-436] VLVQ can be classified under the so-called delayed-decision techniques along with VQ, Tree and trellis coding. Figure 1 exemplifies the distinctives of these delayed decision techniques. Note that VLVQ can be applied, among other things, to the real numbers themselves or, in a (zero-order-) differential-coding perspective, to the real numbers minus the most recently quantized number. Both perspectives are illustrated.



a) Vector Quantization b) Tree coding



c) Trellis coding d) Variable length Vector quantization

Figure 1: Delayed decision coders

FORMAL FRAMEWORK

For the purpose of studying VLVQ a formal framework is required [4]. First a distortion measure must be defined, suitable for variable length vectors.

The set of N vectors used for the quantization is called the dictionary (or codebook) and is noted D_N. Its elements Y_i (i= 1, 2, ..., N) are the codewords and their dimension $\lambda(i)$ is variable: $Y_i = (Y_i^1, Y_i^2, ..., Y_i^{\lambda(i)})$.

Let (X_n) be a sequence of samples, to be encoded. The semi-infinite sequence beginning with the sample X_n is noted $X_n{*}$: $X_n{*} = X_n , X_{n+1} , X_{n+2} \cdots$

Then the distance between $X_n{*}$ and a codeword Y_i is defined by

$$d(x_n{*},Y_i) = (1/\lambda(i)) \ \Sigma_1{}^{\lambda(i)} \ (X_{n+j-1} - Y_i{}^j)^2$$

This is a mean-error-squared distance, which is equivalent to the euclidean distance in the particular case of fixed length vectors. Note that this distortion measure is not a strict mathematical distance.

The first sample that is not coded, $X_{n+\lambda(i)}$, is called the extension sample and is noted X_e .

The encoding method consists of computing all the distances $d(X_n{*}, Y_i)$ for every Y_i in the dictionary, and finding the codeword Y_{i*} that minimizes the distance. The index $i*$ is then transmitted instead of $\lambda(i)$ samples, and the process goes on from sample $X_{n+\lambda(i)}$.

When coding an ergodic sequence of samples, the probability for the codeword Y_i to be chosen is noted $p(i)$. Of course, we have $\Sigma_i \ p(i) = 1$, and the first order quantities can then be defined:

Mean length: $L = \Sigma_i \ p(i)\lambda(i)$ (1)

Mean rate : $R = (\log_2 N)/L$ (2)

Mean squared distortion by sample : $D = (1/L) \ \Sigma_i \ p(i)\lambda(i)D(i)$ (3)

where $D(i)$ is the mathematical expectation for the distance d when Y_i is chosen : $d(i) = E(d/i=i*)$.

It can be written $D = \Sigma_i \ q(i)D(i)$,where $q(i) = p(i)\lambda(i)/L$ represents the probability for a given sample to be coded with Y_i. Thus D effectively measures the average squared distortion by sample.

The above definitions enable the measure of the performances of a VLVQ quantizer. This formalism covers the vector quantization, since in the particular case of a fixed length the definitions for the rate and the distortion turn out to be equivalent to the usual ones.

DESIGN ALGORITHM

A variable length vector quantizer is completely defined by its distortion measure and its dictionary. For an efficient quantization, the dictionary should represent as well as possible the statistical structure of the data to be coded. So it must be determined in an initial training phase. For this purpose, an algorithm for the generation of VLVQ dictionaries has been developped.

It is an iterative algorithm, the size (number of vectors) of the dictionary increasing by one unit at each step. At the initialisation stage, the dictionary is made of a single vector of dimension 1. Then, at each step, a new dictionary is obtained by "splitting" the most probable vector Y_i (the most frequently used) in the dictionary into two new vectors Y_i' and Y_i'' . This splitting may be achieved in three ways:

a) Higher dimension

$Y_i' = Y_i$, $Y_i'' = (Y_i, E(X_e/i))$ where $E(X_e/i)$ is the mathematical expectation for the extension sample X_e when the chosen codeword is Y_i.

b) Same dimension

$Y_i' = Y_i - \varepsilon$, $Y_i'' = Y_i + \varepsilon$ where ε is a vector with small energy.

c) Higher and lower dimension

$Y_i' = Y_i$ without $Y_i^{\lambda(i)}$, $Y_i'' = (Y_i, E(X_e/i))$

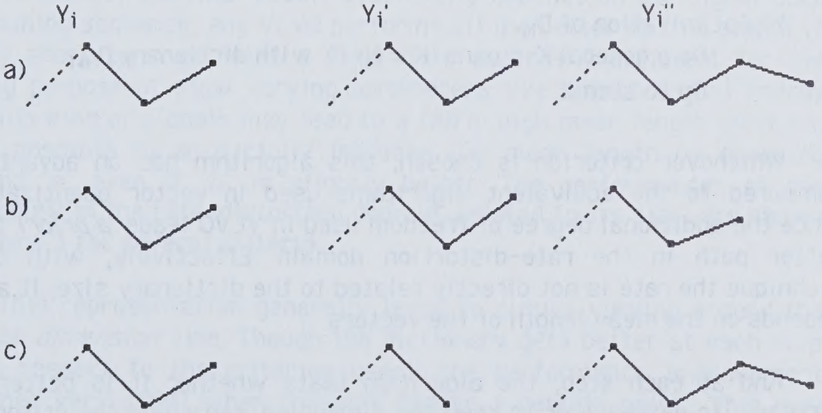


Figure 2: The three splitting methods

These three splitting methods are shown in figure 2. The choice between method a, b or c is made according to a given criterion. So it may be chosen to minimize the distortion as well as to "optimize" the slope in the rate-distortion domain. Besides, it may be useful to assign some constraints to the algorithm (maximal rate for instance).

The algorithm structure is as follow:

1) Initialisation

   D_1 has one vector of dimension 1.

2) Classification

   a) Classify the training sequence (X_n) with VLVQ using dictionary D_N.

   b) For each Y_i in D_N compute $p(i)$, $D(i)$ and $E(X_e/i)$.

   c) Compute the mean length, mean rate and mean distortion using formulas (1), (2) and (3).

3) Test

   If $N > N_0$, stop. Otherwise proceed.

4) Generation of D_{N+1}

   a) Find the most probable Y_i.

   b) Split Y_i with methods a, b and c.

   c) Optimize Y_i' and Y_i'' by doing a K-means $(K = 2)$ over the subset of (X_n) containing the samples coded with Y_i.

   d) Determine the best method according to the chosen criterion.

5) Optimization of D_{N+1}

   Do a general K-means $(K = N+1)$ with dictionary D_{N+1}.

   Go to step 2.

Whichever criterion is chosen, this algorithm has an advantage compared to the equivalent algorithms used in vector quantization since the additional degree of freedom used in VLVQ leads *a priori* to a better path in the rate-distortion domain. Effectively, with this technique the rate is not directly related to the dictionary size; it also depends on the mean length of the vectors.

And at each step, the algorithm tests whether it is better to increase, to decrease or to keep the dimension, regarding the criterion to be optimized. This locally assures its superiority in comparison to an equivalent VQ algorithm. Theoretically speaking, it is not shown that this local superiority is preserved in the final result, but the experiments tend to confirm this hypothesis.

ALGORITHM BEHAVIOR

The algorithm behavior, and thus the final dictionary, essentially depends on the criterion used and on the statistical properties of the data to be coded.

For specific applications the criterion may be the minimum distortion, but generally it should be the optimal slope in the rate-distortion domain, since it gives the lowest rate for a given distortion or, equivalently, the lowest distortion for a given rate.

VLVQ has been tested with different kinds of signals, especially speech signals and some parameters derived from it. A remarkable result of these experiments is that even when the criterion consists of minimizing the distortion and when the training sequence is slightly correlated, the algorithm leads to a final dictionary containing vectors with dimensions greater than one (provided that the dictionary size is large enough). This causes a noticeable reduction in bit rate in relation to scalar quantization, while the mean distortion remains lower or equivalent. For instance a dictionary with 64 vectors, built from differential speech signal, leads approximately to a 25 dB Signal-to-Noise Ratio (which is slightly better than a 6-bits PCM coding). But the mean length is around 1.5 and thus the mean rate is only 4 bits per sample.

Of course, the final result essentially depends on the signal used as training sequence, and VLVQ performs all the better as this signal is highly short-term correlated. Thus it is especially efficient for the coding purpose of slow varying parameters like speech signal energy [5]. This kind of signals may lead to a fairly high mean length. Besides, it is possible to artificially increase the mean length by choosing method "a" even if "b" is slightly better. The performances of the dictionary in the rate-distortion domain, related to its size, are shown in figure 3 for several criteria.

This representation generally leads to curves varying around the *6dB by dimension* line. Though the dictionary gets better at each step (with respect to the criterion used), the performance improvement becomes very small when the size passes a certain point.. This may suggest the interest of seeking an "optimal working point" for the method, which fits the data to be quantized.

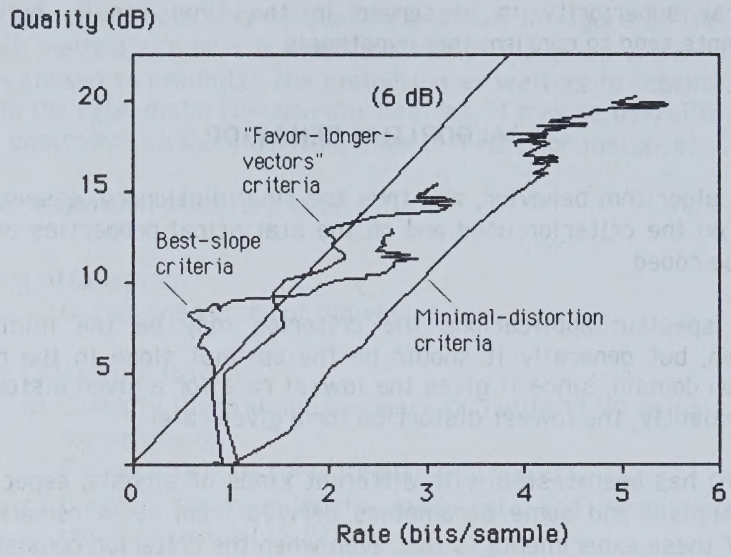


Figure 3: Performances of VLVQ in the rate-distortion domain for several criteria. The codebook size running from 1 to 64.

FUTURE STUDIES

VLVQ appears to be a very promising technique for source coding. Numerous interesting aspects of the method are presented in the paper, as much from a theoretical point of view as from an experimental one. In the theoretical field, the formal framework that has been developped shows that VLVQ generalizes and unifies vector quantization and scalar quantization. In the experimental field, several realistic approaches have been studied,but many potential applications are still to be tested. For instance, it could be interesting to begin with a VQ dictionary and to optimize it with VLVQ, taking advantage of its flexibility.

References:

[1] Adoul, J.P., C. Collin & D. Dalle, "Block Encoding and its Application to Data Compression of PCM Speech", Proceedings of the Canadian Communications & EHV Conference, Montreal 1978, pp. 145-148.

[2] Linde Y., A. Buzo & R.M. Gray, "An Algorithm for Vector Quantizer design", IEEE Trans. Commun., vol. COM-28, nº1, pp. 84-95, Jan. 1980.

[3] Jayant N.S., & P. Noll, "Digital Coding of Waveforms: Principles and Application to Speech and Video", Englewood Cliffs, NJ: Prentice-Hall, 1984.

[4] Adoul, J.P., "Variable Length Vector Quantization", working document, University of Sherbrooke, Canada, May 1983 (unpublished).

[5] Delprat M.., "La quantification vectorielle pour le codage de la paro.le: analyse des méthodes existantes et présentation d'une nouvelle approche", final course report, E.N.S.T. (Paris), France, Jun. 1985.

CONTOUR GAIN VECTOR QUANTIZATION OF GAUSS-MARKOV SOURCES

Kevin T. Malone and Thomas R. Fischer
Department of Electrical Engineering
Texas A&M University
College Station, Texas 77843

ABSTRACT

An information-theory based product code vector quantizer (VQ) is developed for Gaussian autoregressive speech models, and an iterative design algorithm is presented. Full search vector quantizers require the encoder to search the entire output codebook, which for even moderate vector dimensions and low rates is often undesirable. A product code VQ decreases the encoder complexity by decreasing the number of vectors the encoder searches by forming an output codeword as the product of a shape vector and a gain scalar. The contour-gain VQ is a product code VQ where the shape codebook is representative of a high probability contour of constant probability density. The gain codebook is composed of positive scalars which index concentric contours. The performance of the contour-gain VQ is compared to that of the shape-gain VQ by simulating each VQ for several model orders, rates, and dimensions.

INTRODUCTION

The theoretical foundation of vector quantization in source coding commenced with Shannon [1] in 1948. Since then many important contributions to the design and implementation of vector quantizers (VQs) have been made (see e.g. [2-5]). Two major VQ design schemes have evolved. The first design method was developed by Linde, Buzo, and Gray [5] and uses a source representative training sequence to iteratively design a locally optimum VQ. This design algorithm is commonly termed the LBG algorithm. Although it is quite general in formulation and has been applied to various sources [6-9], the LBG algorithm is noted as having two principal drawbacks. First, inherent in the LBG design algorithm is a large computational requirement since it relies on a clustering technique to design the codebook used in encoding. The second difficulty is associated with implementing a VQ designed by means of the algorithm. Due to the unstructured nature of a codebook generated by the LBG algorithm, the encoder complexity generally increases exponentially with rate (for a fixed dimension) or dimension (for a fixed rate). The second VQ design method centers around lattice quanitzers [4] and has been investigated for several sources [10-13]. The structured nature of a lattice quantizer codebook generally serves to reduce encoder complexity.

Tree search algorithms [14,15] and product code VQ's [16,17] have been developed in an attempt to reduce the encoding complexity of VQ's designed using the LBG algorithm. The output codebook of a product code VQ is formed by the Cartesian product of a shape codebook consisting of a finite set of N_1 vectors and a gain codebook consisting of a finite set of N_2 scalars. Thus, the total number of output codewords is $N_1 N_2$. The encoder selects from the shape codebook a codeword representative of the input vector 'shape' and chooses an appropriate gain codeword for the 'gain' of the input vector. The output codeword is the product of the selected gain and contour codewords. Product code VQ's decrease the number of vectors the encoder must search by a factor of $1/N_2$, where N_2 is the number of gain codewords, while only adding N_2 scalar comparisons. Decreasing the number of vectors searched in the encoding process reduces the encoder complexity (which is related to the number of encoder computations).

The shape-gain VQ (SGVQ) [16,17] is probably the best current product code VQ available which uses a clustering algorithm to design the codebooks. The SGVQ approach is to allow the distortion criterion to dictate the structure of the shape codewords. Regardless of the source, the shape codewords of a SGVQ are restricted to lie on a normalized ellipse. In all numerical results this normalized ellipse was simplified to a sphere.

The contour-gain VQ [18] (CGVQ) is also a product code VQ; but, unlike the SGVQ, the CGVQ is motivated by an information theory argument. The CGVQ has been shown to reduce to the SGVQ under certain restrictive assumptions. The original CGVQ design algorithm did not iteratively design the gain codebook, given the shape codebook, and the shape codebook, given the gain codebook. As a result, earlier simulation results obtained for speech waveform coding were somewhat disappointing. In this paper we extend the original CGVQ algorithm to an interative form allowing optimization of the shape codebook for a fixed gain codebook, and vice-versa. Numerical results are presented for first, second, and tenth order Gauss-Markov sources, and a clear improvement over SGVQ performance is demonstrated.

CONTOUR GAIN VECTOR QUANTIZER

The geometric structure of the shape codebook in the CGVQ approach is determined by the source entropy in the following manner. Let $\mathbf{X}$ be a continuous, stationary ensemble with joint probability density function (pdf) $f_{\mathbf{X}}(\mathbf{x}) = f_{\mathbf{X}}(x_1, x_2, \ldots, x_L)$, and let the entropy per degree of freedom [1], h_L, be given by

$$h_L = -\frac{1}{L} \int \cdots \int f_{\mathbf{X}}(\mathbf{x}) \ \log f_{\mathbf{X}}(\mathbf{x})\, d\mathbf{x}. \qquad (1)$$

Consider the large class of sources for which

$$h_L \xrightarrow{\mathbf{P}} h,$$

where h is the source entropy. Shannon [1] demonstrated that for arbitrary δ, $\epsilon > 0$, and for sufficiently large dimension L,

$$\left| \frac{\log f_{\mathbf{X}}(\mathbf{X})}{L} - (-h) \right| < \epsilon \tag{2}$$

for all vectors $\mathbf{X}$, except those on a set of total probability less than δ. Then (1) and (2) imply that, as L gets large, the log of the probability density function of random variable $\mathbf{X}$ approaches a constant, C, given by

$$\frac{\log f_{\mathbf{X}}(\mathbf{X})}{L} \rightarrow C \ = -h. \tag{3}$$

Fischer [13] has shown that for an appropriately large L, $\mathbf{X}$ lies arbitrarily close (in a probabilistic sense) to the high-probability contour specified by (3). Thus, asymptotic in L, we need to consider only the contour of constant probability density $S(L, C)$ given by

$$S(L, C) = \{\mathbf{x} : \frac{\log f_{\mathbf{X}}(\mathbf{x})}{L} = C\}, \tag{4}$$

when assigning VQ output codewords. Most practical VQ applications warrant the use of moderate dimension VQ's. For this case, $\mathbf{X}$ no longer lies as near to the high-probability contour but (3) does provide a good theoretic basis for the CGVQ formulation.

For each nonzero source vector $\mathbf{x}$, let $\beta(\mathbf{x})$ be a positive scaling factor such that

$$\frac{\log f_{\mathbf{X}}(\hat{\mathbf{x}})}{L} = -h_L, \tag{5}$$

where $\hat{\mathbf{x}} = \mathbf{x}/\beta(\mathbf{x})$. Hence, $\hat{\mathbf{x}}$ lies on the high-probability contour determined by the source entropy. Let the 'gain' associated with $\mathbf{x}$ be given by $\beta(\mathbf{x})$. The CGVQ shape codewords are representative of the normalized source vectors $\hat{\mathbf{x}}$, while the gain codewords depict the scaling factors $\beta(\mathbf{x})$. Each gain codeword indexes a particular contour from a set of concentric contours of constant pdf.

As a specific example, consider the Gauss-Markov source. Let $\mathbf{X}$ be a zero-mean, multivariate Gaussian random vector with positive definite covariance matrix $\mathbf{\Lambda}$. Then, the differential entropy of $\mathbf{X}$ [19] is

$$h_L = h_L(\mathbf{x}) = \log\left[\sqrt{2\pi e}|\mathbf{\Lambda}|^{1/L}\right] \tag{6}$$

and the high-probability contour is given by

$$\mathbf{X}^T \mathbf{\Lambda}^{-1} \mathbf{X} = L, \tag{7}$$

where $\mathbf{X}^T$ is the transpose of vector $\mathbf{X}$. The apropriate choice for $\beta(\mathbf{x})$ is then

$$\beta(\mathbf{x}) = \sqrt{\frac{1}{L}\mathbf{x}^T \mathbf{\Lambda}^{-1} \mathbf{x}}. \tag{8}$$

The CGVQ then selects shape codewords based on the contour of high probability (7), and gain codewords based on the gain (8).

CGVQ ITERATIVE DESIGN ALGORITHM

Let the distortion measure used in judging the VQ performance be the standard mean-square error (MSE) given by

$$d = ||\mathbf{x} - \mathbf{y}||_2^2 = \sum_{i=1}^{L} (x_i - y_i)^2, \tag{9}$$

where $\mathbf{x}$ is an L dimensional input vector and $\mathbf{y} = \beta\mathbf{a}$ is the encoded version of $\mathbf{x}$. Let $\{\mathbf{x}_i, i = 1, 2, \cdots, M\}$ be a sequence of L-dimensional vectors which are representative of the source to be encoded. In the original CGVQ design algorithm [18], the contour codebook of N_1 vectors is constructed using the LBG algorithm on a training sequence of normalized vectors $\hat{\mathbf{x}}_i = \mathbf{x}_i/\beta(\mathbf{x}_i)$, where $\beta(\mathbf{x})$ is the appropriate scaling factor for the source. Each $\mathbf{x}_i$ is projected onto the contour codeword closest to $\hat{\mathbf{x}}_i$. This projected vector is denoted by $\mathbf{x}'_i$, and $\beta(\mathbf{x}'_i)$ is computed for $i = 1, \cdots, M$. The LBG algorithm is used to design a locally optimal N_2-level scalar quantizer for the sequence $\{\beta(\mathbf{x}'_i)\}$. Encoding is accomplished by vector quantizing $\hat{\mathbf{x}}$ as the closest contour codeword, and then scalar quantizing $\beta(\mathbf{x}'_i)$ as the closest gain codeword. The output vector is the product of the two codewords.

The new CGVQ design algorithm designs the codebooks in an iterative fashion and also selects the 'best' gain codeword and contour codeword in a manner different from the original algorithm. After designing a contour codebook and a gain codebook, the new algorithm selects the 'best' contour and gain codeword for an input vector $\mathbf{x}$ by finding the $(\beta, \mathbf{a})$ pair from a small neighborhood of output codewords near $\mathbf{x}$ that minimizes d in (9). Given a gain codebook and a contour codebook, the iterative algorithm firsts designs a new gain codebook. For a given contour codeword $\mathbf{a}$, it is easily shown that the β that minimizes d in (9) is given by

$$\beta_{opt} = \frac{\mathbf{x}^T\mathbf{a}}{\mathbf{a}^T\mathbf{a}}. \tag{10}$$

From a training sequence of optimum gains determined by the input vectors and appropriate contour codewords using (10), the new gain codebook is designed via the LBG algorithm. When β in (9) is restricted to be a codeword, d is minimized by selecting the gain codeword from the new gain codebook that is MSE closest to β_{opt}. The new contour codebook is generated by applying the LBG algorithm to a training sequence of $\hat{\mathbf{x}} = \mathbf{x}/\hat{\beta}$, where $\hat{\beta}$ is the new gain codeword MSE closest to β_{opt}. A new $(\beta, \mathbf{a})$ pair that minimizes d is found from a small neighborhood of new output codewords near $\mathbf{x}$, and the design process is repeated. The design process is terminated if the average distortion increases or the decrease in average distortion is minimal.

The iterative CGVQ design algorithm is as follows:

(1) Let $\{\mathbf{x}_i;\ i = 1, \cdots, M\}$ be the sequence of vectors to be encoded, where $\mathbf{x}_i$ is an L-dimensional vector. Let $m = 0$.

(2) Form the sequence $B_0 = \{\beta(\mathbf{x}_i);\ i = 1, \cdots, M\}$, where $\beta(\mathbf{x}_i) = \sqrt{\mathbf{x}_i^T \Lambda^{-1} \mathbf{x}_i}$ and Λ is the covariance matrix of the source to be encoded. Using the LBG algorithm with MSE as the distortion measure and B_0 as the training sequence, design a gain codebook $\mathcal{G}_0$ consisting of N_2 positive scalars. Denote a gain codeword in $\mathcal{G}_0$ by β^0.

(3) Form the sequence $\hat{X}_0 = \{\hat{\mathbf{x}}_i;\ i = 1, \cdots, M\}$, where $\hat{\mathbf{x}}_i = \mathbf{x}_i / \hat{\beta}$, and $\hat{\beta}$ is the gain codeword in $\mathcal{G}_0$ MSE closest to $\beta(\mathbf{x}_i)$. Using the LBG algorithm with MSE as the distortion measure and $\hat{X}$ as the training sequence, design a contour codebook $\mathcal{C}_0$ consisting of N_1 L-dimensional vectors. Denote a contour codeword of $\mathcal{C}_0$ by $\mathbf{a}^0$.

(4) For each $\mathbf{x}_i$, do the following. Let $\hat{\beta}_1$ and $\hat{\beta}_2$ be the two MSE closest gain codewords of $\mathcal{G}_0$ to $\beta(\mathbf{x}_i)$, and $\hat{\mathbf{a}}_1, \hat{\mathbf{a}}_2, \cdots, \hat{\mathbf{a}}_K$ be the K contour codewords of $\mathcal{C}_0$ that yield the K largest values of $\mathbf{x}_i^T \mathbf{a}^0 / ||\mathbf{a}^0||_2$. For β_i^0 equal to $\hat{\beta}_1$ or $\hat{\beta}_2$, and $\mathbf{a}_i^0 = \hat{\mathbf{a}}_k, k = 1, \cdots, K$; determine the pair $(\beta_i^0, \mathbf{a}_i^0)$ that minimizes $d_i = ||\mathbf{x}_i - \beta_i^0 \mathbf{a}_i^0||_2^2$. Compute the average distortion $D^0 = E[d_i]$.

(5) Let $m = m + 1$. Form the sequence $B_m = \{\beta^m_{opt,i}(\mathbf{x}_i, \mathbf{a}_i^{m-1});\ i = 1, \cdots, M\}$ where $\beta^m_{opt,i}(\mathbf{x}_i, \mathbf{a}_i^{m-1}) = \mathbf{x}_i^T \mathbf{a}_i^{m-1} / ||\mathbf{a}_i^{m-1}||_2^2$. Using the LBG algorithm with MSE as the distortion measure and B_m as the training sequence, design a new gain codebook $\mathcal{G}_m$ consisting of N_2 positive scalars. Denote a gain codeword of $\mathcal{G}_m$ by β^m.

(6) For each $\mathbf{x}_i$, let $\hat{\beta}_1$ and $\hat{\beta}_2$ denote the two gain codewords of $\mathcal{G}_m$ closest to $\beta^m_{opt,i}(\mathbf{x}_i, \mathbf{a}_i^{m-1})$. Denote the $\hat{\beta}_i, i = 1, 2;$ that minimizes $||\mathbf{x}_i - \hat{\beta}_j \mathbf{a}^{m-1}||_2^2$ by $\hat{\beta}_i^m$.

(7) Form the sequence $\hat{X}_m = \{\hat{\mathbf{x}}_i;\ i = 1, \cdots, M\}$, where $\hat{\mathbf{x}}_i = \mathbf{x}_i / \hat{\beta}_i^m$. Using the LBG algorithm with MSE as the distortion measure and $\hat{X}_m$ as the training sequence, design a new contour codebook $\mathcal{C}_m$ consisting of N_1 L-dimension vectors. Denote a contour codeword of $\mathcal{C}_m$ by $\mathbf{a}^m$.

(8) For each $\mathbf{x}_i \epsilon X$, do the following. Find the K contour codewords of $\mathcal{C}_m$ that yield the K largest values of $\mathbf{x}_i^T \mathbf{a}^m / ||\mathbf{a}^m||_2$. Denote these contour codewords by $\hat{\mathbf{a}}_1, \hat{\mathbf{a}}_2, \cdots, \hat{\mathbf{a}}_K$, where $\mathbf{x}_i^T \hat{\mathbf{a}}_1 / ||\hat{\mathbf{a}}_1||_2 \geq \mathbf{x}_i^T \hat{\mathbf{a}}_j / ||\hat{\mathbf{a}}_j||_2$, for $j = 2, 3, \cdots, K$. Let $\hat{\beta}_1$ and $\hat{\beta}_2$ denote the two gain codewords in $\mathcal{G}_m$ that are MSE closest to $\mathbf{x}_i^T \hat{\mathbf{a}}_1 / ||\hat{\mathbf{a}}_1||_2^2$. For β_i^m equal to $\hat{\beta}_1$ or $\hat{\beta}_2$ and $\mathbf{a}_i^m = \hat{\mathbf{a}}_k$, $k = 1, 2, \cdots, K$; determine the pair $(\beta_i^m, \mathbf{a}_i^m)$ that minimizes $d_i = ||\mathbf{x}_i - \beta_i^m \mathbf{a}_i^m||_2^2$.

(9) Compute the average distortion $D^m = E[d_i]$. Let $\delta = (D^m - D^{m-1}) / D^{m-1}$ be the termination criteria. If $\delta > \alpha$ then go to step (5). If $0 < \delta < \alpha$ then output codebooks $\mathcal{G}_m$ and $\mathcal{C}_m$ and stop. If $\delta < 0$ then output codebooks $\mathcal{G}_{m-1}$ and $\mathcal{C}_{m-1}$ and stop.

Let $\mathcal{G}$ and $\mathcal{C}$ be the gain and contour codebooks design by the CGVQ iterative algorithm. An input vector $\mathbf{x}$ is encoded by first finding the K contour codewords of $\mathcal{C}$ that yield the K largest values of $\mathbf{x}^T\mathbf{a}/||\mathbf{a}||_2$, where $\mathbf{a}\epsilon\mathcal{C}$. Denote these contour codewords by $\hat{\mathbf{a}}_1, \hat{\mathbf{a}}_2, \cdots, \hat{\mathbf{a}}_K$, where $\mathbf{x}^T\hat{\mathbf{a}}_1/||\hat{\mathbf{a}}_1||_2 \geq \mathbf{x}^T\hat{\mathbf{a}}_j/||\hat{\mathbf{a}}_j||_2$, for $j = 2, 3, \cdots, K$. Let $\hat{\beta}_1$ and $\hat{\beta}_2$ denote the two gain codewords in $\mathcal{G}$ that are MSE closest to $\mathbf{x}^T\hat{\mathbf{a}}_1/||\hat{\mathbf{a}}_1||_2^2$. For $\tilde{\beta}$ equal to $\hat{\beta}_1$ or $\hat{\beta}_2$ and $\tilde{\mathbf{a}} = \hat{\mathbf{a}}_k$, $k = 1, 2, \cdots, K$; determine the pair $(\tilde{\beta}_i, \mathbf{a})$ that minimizes $d = ||\mathbf{x}_i - \tilde{\beta}\tilde{\mathbf{a}}||_2^2$.

CGVQ PERFORMANCE

Shape-gain VQ's and contour-gain VQ's were designed and simulated for first, second, and tenth order Gaussian autoregressive (AR) speech models. The SGVQ's were designed using the individually optimized (I.O.) algorithm outlined in [17]. There is little difference in the performance of all three SGVQ design algorithms in [17]; the I.O. algorithm was selected on the basis of the recommendation in the paper.

The initial codebook for the LBG algorithm was comprised of members of the training sequence. The termination threshold for the LBG algorithm was $\epsilon = 0.0001$. The partition regions for the CGVQ's were based on Euclidean distance from the centroid, and the region centroid was the mean of the partition region members. The partition regions and centroids used in the SGVQ design were as specified in [17]. The value of α used in the CGVQ iterative design algorithm was $\alpha = 0.02$.

For both VQ's, the performance of the VQ was found when the input sequence to the VQ was the design sequence and also for a test sequence input. Each sequence consisted of approximately 116,000 samples. The normalized MSE performance per dimension of the SGVQ and CGVQ for several dismensions, L, and rates, R, are shown in Tables 1 and 2. The CGVQ performance is between 2% and 14% better than the SGVQ performance for a similar R, L, and AR order. No attempt was made to find the (N_1, N_2) pair that minimized the CGVQ distortion. Other combinations of N_1 and N_2 could possibly yield lower distortions for the CGVQ. When $N_1 N_2$ is less than 2^{RL}, the CGVQ performance is improved if an input vector $\mathbf{x}$ is quantized as the all zero vector when d in (9) is greater than $||\mathbf{x}||_2^2$. The total number of output codewords is then $N_1 N_2 + 1$. The increase in performance was very slight for the case when $L = 4$ and $R = 1$, and the increase was negligible for the other two cases. The SGVQ algorithm can be modified in the same mannner, but was not modified for these simulations. The CGVQ performance for the $R = 1$, $L = 4$ case was also improved by removing low energy vectors from the design sequence before entering the iterative algorithm. The threshold value, T, is chosen to be less than the estimated MSE. A threshold value of $T = 34,000$ was used in the CGVG design when L was four. No thresholding was performed when the vector dimension was greater than four. The MSE values listed in Table 2 are those obtained using a threshold and zero vector where applicable.

CONCLUSION

The CGVQ MSE performance for the three AR models has been shown to be uniformly better than the SGVQ performance for the cases investigated. The CGVQ design algorithm is more complex that that for the SGVQ, while the CGVQ encoder is only slightly more complex than the SGVQ encoder.

ACKNOWLEDGEMENT

This research was supported in part by the Louise Lindsey Merrick Graduate Fellowship from the Texas A&M University College of Engineering.

TABLE I.
SGVQ MSE Performance

AR Order	Rate R	Dim. L	Shape Vectors N_1	Gain Levels N_2	Normalized MSE In Sequence	Normalized MSE Test Sequence
1	1	4	4	4	.14209	.14013
1	1	4	5	3	.14903	.14710
1	1	4	7	3	.17730	.17296
1	1	7	21	6	.11632	.11504
1	1	7	25	5	.11538	.11455
1	1	7	32	4	.11638	.11610
1	2	6	455	9	.03440	.03883
1	2	6	512	8	.03426	.03889
1	2	6	585	7	.03453	.03927
2	1	4	4	4	.16387	.16411
2	1	4	5	3	.14999	.14953
2	1	4	7	2	.15078	.14976
2	1	7	25	5	.08794	.08898
2	1	7	32	4	.08429	.08488
2	1	7	42	3	.08784	.08723
2	2	6	372	11	.01821	.01997
2	2	6	409	10	.01799	.01975
2	2	6	455	9	.01815	.01996
10	1	4	4	4	.17406	.17578
10	1	4	5	3	.16174	.16441
10	1	4	7	2	.16523	.16431
10	1	7	25	5	.10684	.10742
10	1	7	32	4	.10209	.10332
10	1	7	42	3	.10491	.10495
10	2	6	455	9	.02622	.02964
10	2	6	512	8	.02611	.02967
10	2	6	585	7	.02624	.03006

TABLE II.
CGVQ MSE Performance

AR Order	Rate R	Dim. L	Contour Vectors N_1	Gain Levels N_2	K Parameter	Normalized MSE In Sequence	Normalized MSE Test Sequence
1	1	4	7	2	3	.13325	.13248
1	1	7	42	3	4	.10323	.10385
1	2	6	455	9	4	.03262	.03643
2	1	4	7	2	3	.12698	.12650
2	1	7	42	3	4	.08003	.08094
2	2	6	455	9	4	.01713	.01872
10	1	4	7	2	3	.14130	.14111
10	1	7	42	3	4	.09834	.09943
10	2	6	455	9	4	.02550	.02824

REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Systems Tech. J.*, vol. 27, pp. 379–423, 623–656, 1948.

[2] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE National Conv. Rec.*, part 4, pp. 142–163, 1959.

[3] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inform. Theory*, vol. IT–25, pp. 373–380, July 1979.

[4] A. Gersho, "On the structure of vector quantizers," *IEEE Trans. Inform. Theory*, vol. IT–28, pp. 157–166, March 1982.

[5] Y. L. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM–28, pp. 84–95, Jan. 1980.

[6] R. M. Gray, and Y. Linde, "Vector quantizers and predictive quantizers for Gauss-Markov sources," *IEEE Trans. Commun.*, vol. COM–30, pp. 381–389, Feb. 1982.

[7] H. Abut, R. M. Gray, and G. Rebolledo, "Vector quantization of speech and speech-like waveforms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP–30, pp. 423–435, June 1982.

[8] A. Buzo, A. H. Gray, Jr., R. M. Gray, and J. D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP–28, pp. 562–574, Oct. 1980.

[9] V. Cuperman and A. Gersho, "Adaptive differential vector coding of speech," *Proc. Globecom '82*, pp. 1092–1096, Nov. 1982.

[10] J. H. Conway and N. J. A. Sloane, "Voronoi regions of lattices, second moments of polytopes, and quantization," *IEEE Trans. Inform. Theory*, vol. IT–28, pp. 211–226, March 1982.

[11] J. H. Conway and N. J. A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inform. Theory*, vol. IT–28, pp. 227–232, March 1982.

[12] J. H. Conway and N. J. A. Sloane, "A fast encoding method for lattice codes and quantizers," *IEEE Trans. Inform. Theory*, vol. IT–29, pp. 820–824, Nov. 1983.

[13] T. R. Fischer, "A pyramid vector quantizer," *IEEE Trans. Inform. Theory*, vol. IT–32, pp. 568–583, July 1986..

[14] A. Gersho and D.-Y. Cheng, "Fast nearest-neighbor search for Euclidean codes," *Abstracts of the 1983 IEEE International Symposium on Information Theory,* p. 88, Ste. Jovite, Canada, Sept. 1983

[15] R. M. Gray and H. Abut, "Full-search and tree-search vector quantization of speech waveforms," in *Proc. Intl. Conf. on Acoust., Speech, Signal Processing*, pp. 593–596, Paris, April 1982.

[16] M. J. Sabin and R. M. Gray, "Product code vector quantizers for speech waveform coding," in *Conf. Rec. of the 1982 IEEE Global Comunications Conf.*, pp. E6.5.1–E6.5.5, Miami, FL, 1982.

[17] M. J. Sabin and R. M. Gray, "Product code vector quantizers for waveform and voice coding," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP–32, pp. 474–488, June 1984.

[18] T. R. Fischer and K. T. Malone, "Contour vectour quantization and waveform coding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 1707–1710, Tampa, FL, March 1985.

[19] T. Berger, *Rate Distortion Theory.* Englewood Cliffs, NJ: Prentice-Hall Inc., 1971.

ON "SOFT" BIT ALLOCATION

A.M. Bruckstein

Department of Electrical Engineering
TECHNION, Israel Institute of Technology
Haifa, 32000, Israel

ABSTRACT

If a random variable X with variance σ^2 is quantized optimally, being mapped into s discrete levels, the quantization error is roughly proportional to σ^2/s^2. In many applications in speech coding and image digitization we are given sets of random variables and we have to quantize them in some way so as to represent their realizations with as few discretized levels (bits) as possible. This has to be done while also minimizing the total quantization error. The quantization level or bit allocation process should therefore be the result of a compromise between the total discretization error and the number of bits used to represent the realizations of random vectors. This note presents a general solution of this problem, extending the classical bit allocation methods in which a fixed, prescribed number of bits have to be allocated to minimize the quantization error. The general soft bit allocation process is useful in designing variable rate (adadptive) coders as opposed to the classical bit allocation procedures that were devised for information transmission over communication channels requiring constant rate encoding.

SUMMARY

Suppose we are given M random variables, $X_1, X_2, \ldots ,X_M$ with variances $\sigma_1^2 \geq \sigma_2^2 \geq \sigma_3 \geq \cdots \geq \sigma_M$, and we separately quantize them using s_i quantization levels for X_i. Then the total quantization error, with optimal quantization, is approximately proportional to

$$E_Q = \sum_{i=1}^{M} \sigma_i^2/s_i^2 \tag{1}$$

and the total number of bits needed to code the $\prod_{i=1}^{M} s_i$ possible discretized levels for the vector $(X_1, X_2, \ldots ,X_M)$ is

$$B_T = \log_2 \prod_{i=1}^{M} s_i = \sum_{i=1}^{M} \log_2 s_i \tag{2}$$

We note that the quantization error for uniformly distributed random variables is exactly proportional to the expression given by (1). The classical bit allocation or integer quantization level allocation problem is to choose the natural numbers s_i so as to meet the requirement that $B_T = B$ (a certain predetermined integer), while minimizing E_Q. The motivation for such a bit allocation process was a requirement to transmit information on successive realizations of the vector $(X_1, X_2, \ldots, X_M)$ over a *fixed capacity* communication channel. This process however does not allow any flexibility in either reducing the bit rate when the quantization error is smaller than required or in increasing it when it becomes temporarily important to improve the precision of describing $X_1, X_2, \ldots, X_M$. With present technology it is not a problem to dynamically allocate channel capacity in order to accomodate a source with variable bit generation rate, and improved performance certainly results from adaptation of the quantization process to slowly varying statistical properties of the sources or to changing demands on precision.

We propose a "soft" bit allocation procedure. It will be assumed that two functions, $\mathbb{C}_Q(\cdot)$ and $\mathbb{C}_D(\cdot)$ that measure the cost incurred by quantization error and by bit usage respectively, are given and we shall determine an integer quantization level allocation, $\{s_i\}$, that minimizes their sum, for a source vector characterized by the given set of variances $\{\sigma_1^2, \sigma_2^2, \ldots, \sigma_M^2\}$.

The total cost to be minimized, over all feasible allocations $\{s_i\}$, is therefore the function

$$\mathbb{C}(\{s_i\}) = \mathbb{C}_Q\left[\sum_{i=1}^{M} \sigma_i^2/s_i^2\right] + \mathbb{C}_D\left[\sum_{i=1}^{M} \log_2 s_i\right] \tag{3}$$

where $\mathbb{C}_Q$ and $\mathbb{C}_D$ are monotone increasing and smooth (differentiable) functions. The minimization of this function has to be carried out under the contraint that s_i are natural numbers greater than one.

This method generalizes the classical bit allocation procedure which requires B bits to be allocated to M variables so as to minimize the quantization error. As a particular case of this general approach, we also solve the dual problem of assigning a variable number of bits so as to enforce an upper bound on the quantization error. For further discussion and references to related work see [1].

[1] A. M. BRUCKSTEIN, "On soft bit allocation", to appear in IEEE Trans. on Acoustics, Speech and Signal Processing, 1986/87.

ARRAY CALIBRATION IN A CORRELATED SIGNAL ENVIRONMENT *

TIANHU LEI and C. NELSON DORNY
The Moore School of Electrical Engineering
University of Pennsylvania
Philadelphia, PA 19104

Previous work [1 - 5] in calibration of large antenna arrays uses phase multilateration to locate a set of far - field point sources together with the element positions. This work assumes the signal environment to be uncorrelated and requires that the 2π ambiguity associated with phase measurement be eliminated prior to the calibration procedure. This article proposes a new approach to array calibration which can be used in a general correlated signal environment and is not hampered by the 2π ambiguity associated with phase measurement. The calibration procedure is designed for the near - field, but extends easily to the far - field.

Sinusoidal signals are emitted by (or reflected from) point sources, and the phases of these signals are measured at the elements. The sample covariance matrix of these phase measurements is used to estimate the parameters of the probability distribution function of the set of measured phases, and represents a statistical description of the overall signal environment. The estimates of the means of the measured phases are affected by the estimated element and source positions. A ML estimation procedure is used to produce optimum estimates of the means of the phases, and thereby, optimum estimates of the positions [6 - 7].

The ML estimation is implemented by a steepest - ascent algorithm (Self - Survey) which tolerates essentially unlimited uncertainty in the initial estimates of the positions of sources and elements, despite the multimodal nature of the likelihood functoin owing to 2π ambiguity in phase measurement. That is, global convergence is assumed. Because of the combination of phase error and 2π ambiguity, it is theoretically possible for the global maximum to be a false (ambiguous) solustion. However, the Self - Survey algorithm uses a stopping criterion, based on the concept of residual phase, which is a measure of the quality of the resulting calibrated array as an imaging device. The probability of misclassification (termination at an ambiguous solution) is shown to be negligible even for low SNR [8].

A tolerance theory gives measures of the levels of the errors in the final estimates of the positions and provides valuabel design information for imaging systems. It is shown that the variance in the array phase errors owing to miscalibration is proportional to $[1 + (1 - P)\rho_{max}]/SNR_e$, where ρ is the signal correlation coefficient, P is the minimum number of source - to - element channels used during Self - Survey (in the order 20), and SNR_e is the effective SNR. Simulation examples and testing via experimental data verify the theory and the effectiveness of the algorithm [8]. A hypothesis test for recognition of adequate "point" sources is also given [9].

---

* This work is principally supported by White Sands Missile Range Contract $DAAD$ $07 - 84R - 0081$ through Subcontract #110A from Interspec Inc.

REFERENCES

1. C. N. Dorny, “A Self-Survey Technique for Self-Cohering of antenna System,” *IEEE Trans. Antennas Propag.*, vol. **AP-26**, Nov. 1978.
2. E. A. Lee, “A Generalized Self-Survey Technique for Self-Cohering of a Large Array,” Ph.D. dissertation, University of Pennsylvania, Philadelphia, 1981.
3. S. A. Kassam and Y. Shamash, “Location of Random Array Elements Using Phase Measurement”, *8th Annual Modeling and Simulation Conference*, Pittsburgh, Pennsylvania, April 1977.
4. J. Tsao, “A Position Error Correction Algorithm and a Revised Clean Technique for Random Thinned Array Imaging Systems,” Ph.d. dissertation, University of Pennsylvania, Philadelphia, 1983.
5. Y. Rockah, H. Messer, and P. M. Schultheiss “Source Localization with an Array Subject to Uncertainty in Sensor Phase,” *The 20th Annual Conference on Information and Systems,* Princeton University, Princeton, New Jewsey, March 1986.
6. T. Lei and C. N. Dorny, “Necessary and Sufficient Conditions for Maximization of Multidimensional Gaussian PDF,” *The 20th Annual Conference on Information and Systems.* Princeton University, Princeton New Jersey, March 1986.
7. T. W. Anderson, *An Introduction to Multivariate Statistical Analysis,* John Wiley & Sons Inc., New York, 1958.
8. Tianhu Lei, “Array Calibration in a Correlated Signal Environment,” Ph.d. dissertation, University of Pennsylvania, Philadelphia, 1986.
9. T. Lei and C. N. Dorny, “Recognition of Point Sources,” *IEEE Montech' 86 Conference on Antenna and Communications,* Montreal, Quebec, Canada, Sept. 29 - Oct. 1, 1986.

CAPACITY REGION OF GAUSSIAN CDMA CHANNELS: THE SYMBOL-SYNCHRONOUS CASE

SERGIO VERDU

Department of Electrical Engineering
Princeton University
Princeton, NJ 08544

Abstract: *The $K-$user capacity region of the white Gaussian Code-Division Multiple-Access channel is found in terms of the crosscorrelations between the assigned signature waveforms and their signal-to-noise ratios under the assumption of power-constrained inputs. Also, the conventional Gaussian multiple-access channel is studied under the assumption of amplitude-constrained inputs and its finite- and infinite-user capacity regions are found under various assumptions on the allowable amplitudes.*

1. Introduction

A major multiaccessing mode in radio networks and other common-channel communication networks is Code-Division Multiple-Access (CDMA). In this technique, each transmitter is assigned a fixed, distinct signature waveform which is used to modulate a stream of digital information as in single-user communication. In this way, the transmitters are not required to cooperate in order to schedule their transmission times, and all of them are allowed to transmit simultaneously and independently. Then, the receiver demodulates the transmitted messages upon observing the sum of the transmitted signals embedded in noise. Ideally, if the assigned waveforms were orthogonal, then (assuming the additive noise is white and Gaussian) the multiuser channel decouples into single-user channels. However, oftentimes in practice nonorthogonal waveforms are assigned because of bandwidth or complexity limitations or because of lack of timing synchronism among the users. In this situation the performance limits of the CDMA system depend on the degree of similarity (i.e., crosscorrelations) between the assigned waveforms. Typically these are pseudo-noise signature sequences operating in high-bandwidth channels. In this paper, the capacity region (i.e., the set of information rates at which reliable communication is possible) of the CDMA channel is found as a function of the crosscorrelations between the assigned waveforms and their signal-to-noise ratios. Therefore, the set of assigned waveforms is fixed and assumed to be part of the channel. But the signalling waveforms are indeed chosen by the designer, so the question arises as to whether such a capacity region really quantifies a fundamental limit. The answer is that analysis is a prerequisite to design; once the maximum throughput is known as a function of the crosscorrelations, the signature waveforms can be optimized under the specific constraints of the problem. Bandwidth is not the only or, often, the major constraint facing the designer of the CDMA system; frequently, the degree of crosscorrelation achievable under the specific limitations of complexity, bandwidth, number of chips-per-symbol, etc. is a harder, more definite constraint.

The chief assumptions under which the capacity region of the CDMA channel is obtained in this paper are a) the receiver observes the sum of transmitted signals imbedded in additive white Gaussian noise, b) the users are symbol-synchronous, i.e., their symbol-epochs coincide at the receiver, and c) the modulation is linear. No assumptions are placed on the assigned waveforms, other than their energy is finite and their duration is the same for all users and finite. Hence, the multiple-access channel under consideration is given by

$$r(t) = \sum_{i} \sum_{k=1}^{K} b_k(i)\, s_k(t - iT) + n(t). \tag{1.1}$$

where K is the number of users, $\{b_k(i)\}_i$ is the sequence of symbols transmitted by the $k-$th user, and $s_k(t)$ ($= 0$, if $t \notin [0,T]$) is the fixed waveform assigned to the $k-th$ user. It should be noted that

---

This work was partially supported by the National Science Foundation under Grant ECS-8504752

the assumption of symbol-synchronism is crucial for the results of this paper and their proofs. Symbol-asynchronous channels are studied in [1].

The conventional Gaussian multiple-access channel is

$$Y = \sum_{k=1}^{K} X_k + N \tag{1.2}$$

where N is $N(0,\sigma^2)$. Under the assumption that the power of the $k-th$ input is restricted to w_k, Cover [2] and Wyner [3] showed that the capacity region of (1.2) is

$$\bigcap_{J \subset \{1, \cdots K\}} \{(R_1, \cdots R_K) : 0 \le \sum_{i \in J} R_i \le \frac{1}{2} \log [\, 1 + \sum_{i \in J} \frac{w_i}{\sigma^2}\,]\, \} \tag{1.3}$$

The conventional Gaussian multiple-access channel does not encompass (1.1) unless all users are assigned the same waveform. As it is shown in Section 2, a multidimensional generalization of (1.2) (or equivalently a scalar channel with memory) is required in order to analyze (1.1). The case where all users are assigned the same waveform is also of interest in connexion with the CDMA channel, because it models the extreme situation where the signature of each user consists exclusively of its assigned codebook. In a Direct-Sequence Spread-Spectrum system, this would correspond to assigning the same chip waveform to each user and to letting each chip be modulated by a different symbol. Although, admittedly, such a system can hardly be categorized as Direct-Sequence Spread-Spectrum since there are no signature sequences and no bandwidth expansion, it is still useful to analyze its capacity and compare it to the assigned waveform channel, especially in the case of amplitude-constrained inputs (Section 3).

Except for a recent paper by Viterbi [4], previous works related to the capacity of CDMA channels (e.g., [5-7]) have all analyzed single-user demodulation approximating the multiple-access interference by a white Gaussian process. Viterbi [4] has examined the infinite-user behavior of the conventional multiple-access channel (1.2) under amplitude constrained inputs, and has shown the achievability (by the "successive cancellation" technique) of the same rate-sum that would be attained by variance constrained users. In Section 3, the capacity region of this infinite-user channel is found.

2. Capacity with Power-Constrained Inputs

In this section, the set of $K-$user rates $(R_1, \cdots, R_K)$ for which reliable communication is possible is found as a function of the crosscorrelations between the assigned signals, the energy of each user and the background white Gaussian noise spectral level. It is assumed that each assigned waveform is linearly modulated by a sequence of real-valued amplitudes that satisfy an average power constraint:

$$\frac{1}{n} \sum_{i=1}^{n} E[b_k^2(i)] \le 1 \qquad k = 1, \cdots, K \tag{2.1}$$

for each block of n transmitted symbols. There is no loss of generality in (2.1) since the energy of the assigned waveforms has not been assumed to be normalized.

Recall that the receiver observes the sum of modulated signals embedded in additive white Gaussian noise with spectral density equal to σ^2, i.e.,

$$r(t) = \sum_{i=1}^{n} \sum_{k=1}^{K} b_k(i)\, s_k(t - iT) + n(t). \tag{2.2}$$

The capacity region of this channel is given by the following result.

Proposition 1: Let $\mathbf{H}$ denote the crosscorrelation matrix of the signal constellation $\{s_1(t), \cdots, s_K(t)\}$, i.e.,

$$H_{ij} = \int_0^T s_i(t)\, s_j(t)\, dt.$$

Then, the capacity region of the Gaussian multiple-access channel (2.2) with power constrained inputs is equal to

$$\bigcap_{J \subset \{1, \cdots K\}} \{(R_1, \cdots R_K) : 0 \le \sum_{i \in J} R_i \le \frac{1}{2} \log [\, \det (\, \mathbf{I}_{|J|} + \sigma^{-2}\, \mathbf{H}_J\,)\,]\, \} \tag{2.3}$$

where $\mathbf{H}_J$ is the $|J| \times |J|$ submatrix of $\mathbf{H}$ obtained by striking out the rows and columns of $\mathbf{H}$ whose indices do not belong to the set J.

Proof: The first step is to derive a discrete-time model for the multiple-access channel (2.2). Since each input is already a discrete-time process (namely, the sequence of symbols that modulate the waveform assigned to the corresponding user), it suffices to find a sequence of observables that are sufficient statistics for the input sequence $\{\mathbf{b}(i) = [b_1(i), \cdots b_K(i)]^T, i = 1, \cdots n\}$. To that end, suppose that the finite-energy signals $\{s_1(t), \cdots s_K(t)\}$ span an M-dimensional subspace of $L_2[0,T]$, of which $\{\phi_1(t), \cdots \phi_M(t)\}$ is an orthogonal basis. Then, since the additive noise process in (2.2) is white and Gaussian, it is well-known that $\{\mathbf{y}(i) = [y_1(i), \cdots y_K(i)]^T, i = 1, \cdots n\}$ is a sequence of sufficient statistics if

$$y_k(i) = \int_{iT}^{iT+T} r(t)\, \phi_k(t - iT)\, dt. \tag{2.4}$$

Therefore, an equivalent discrete-time model is

$$\mathbf{y}(i) = \mathbf{A}\,\mathbf{b}(i) + \mathbf{n}(i) \tag{2.5}$$

where $\mathbf{A}$ is an $M \times K$ matrix such that

$$s_j(t) = \sum_{i=1}^{M} A_{ij}\, \phi_i(t), \tag{2.6}$$

and $\mathbf{n}(i)$ is a zero-mean M-dimensional Gaussian process with covariance

$$E[\mathbf{n}(i)\, \mathbf{n}(j)^T] = \sigma^2\, \delta_{ij}\, I_M. \tag{2.7}$$

Now, since the discrete-time channel in (2.5) is memoryless its capacity region is equal to the convex hull of the regions (e.g. [8]),

$$\bigcap_{J \subset \{1, \cdots K\}} \{(R_1, \cdots R_K):\ 0 \le \sum_{i \in J} R_i \le I(\{X_i\}_{i \in J};\, \mathbf{Y} \mid \{X_i\}_{i \notin J})\} \tag{2.8}$$

over all product distributions of $\mathbf{X} = (X_1, \cdots, X_K)$ on R^K such that $E[X_k^2] \le 1, k = 1, \cdots, K$ and with

$$\mathbf{Y} = \mathbf{A}\,\mathbf{X} + \mathbf{N} \tag{2.9}$$

where $\mathbf{N}$ is a zero mean Gaussian random K-vector with covariance matrix $\sigma^2\, \mathbf{I}_K$.

Analogously to the case of the conventional one-dimensional Gaussian multiple-access channel with power-constrained inputs [3], here we are fortunate that there is a product input distribution whose rate region includes that of any other input distribution. This property is a consequence of the assumption that the users are symbol-synchronous and does not hold otherwise [1].

The proof of (2.3) will follow by showing that for every $J \subset \{1, \cdots K\}$ and every admissible input distribution

$$I(\{X_i\}_{i \in J};\, \mathbf{Y} \mid \{X_i\}_{i \notin J}) \le \frac{1}{2} \log\left[\det\left(I_{|J|} + \sigma^{-2}\mathbf{H}_J\right)\right], \tag{2.10}$$

with equality if the inputs $(X_1, \cdots X_K)$ are independent standard Gaussian random variables. Letting $h(F)$ denote the differential entropy of the probability distribution function F, we can write the mutual information in (2.10) as

$$I(\{X_i\}_{i \in J};\, \mathbf{Y} \mid \{X_i\}_{i \notin J}) = h(F_{\mathbf{Y}_J}) - h(F_{\mathbf{N}}), \tag{2.11}$$

where $F_{\mathbf{Y}_J}$ is the distribution of $\mathbf{N} + \sum_{i \in J} X_i$ and $F_{\mathbf{N}}$ is the distribution of $\mathbf{N}$.

Letting $\Xi_J = diag\,(1\{1 \in J\}, \cdots 1\{K \in J\})$, the differential entropies in (2.11) satisfy the following relationships (e.g. [9,p.44]),

$$h(F_{\mathbf{N}}) = \frac{M}{2} \log[2\pi e] + \log[\sigma] \tag{2.12}$$

$$h(F_{\mathbf{Y}_J}) \le \frac{M}{2} \log[2\pi e] + \frac{1}{2} \log\left[\det\left(\sigma^2 I_M + \mathbf{A}\, \Xi_J\, cov(\mathbf{X})\, \Xi_J\, \mathbf{A}^T\right)\right] \tag{2.13}$$

with equality in (2.13) if X is a Gaussian K-vector. But since the constraints on the admissible inputs only affect $cov(\mathbf{X})$ it follows that it is enough to restrict attention to Gaussian inputs. Moreover, since the identity is the maximal element of the set of admissible input covariances, it follows that (e.g., [10,p.141])

$$\det\,(\sigma^2\mathbf{I}_M + \mathbf{A}\;\Xi_J\;cov(\mathbf{X})\;\Xi_J\;\mathbf{A}^T) \;\leq\; \det\,(\sigma^2\mathbf{I}_M + \mathbf{A}\;\Xi_J\;\mathbf{A}^T) \tag{2.14}$$

for any subset $J \subset \{1, \cdots K\}$. Consequently, as was to be expected the rate region defined by the mutual informations achieved by independent inputs with maximum allowable variance contains the rate region achieved by any other admissible distribution.

To complete the proof it remains to show that

$$\det\,(\mathbf{I}_M + \sigma^{-2}\mathbf{A}\;\Xi_J\;\mathbf{A}^T) \;=\; \det\,(\mathbf{I}_{|J|} + \sigma^{-2}\mathbf{H}_J) \tag{2.15}$$

On the one hand, it is easy to check by row/column permutations of $\Xi_J\;\mathbf{H}\;\Xi_J$ that

$$\det\,(\mathbf{I}_{|J|} + \sigma^{-2}\mathbf{H}_J) \;=\; \det\,(\mathbf{I}_K + \sigma^{-2}\Xi_J\;\mathbf{H}\;\Xi_J). \tag{2.16}$$

And on the other hand,

$$\begin{aligned}\det\,(\mathbf{I}_M + \sigma^{-2}\mathbf{A}\;\Xi_J\;\mathbf{A}^T) &= \det\,(\mathbf{I}_M + \sigma^{-2}\mathbf{A}\;\Xi_J\;\Xi_J\;\mathbf{A}^T)\\ &= \det\,(\mathbf{I}_K + \sigma^{-2}\Xi_J\;\mathbf{A}^T\mathbf{A}\;\Xi_J)\end{aligned}$$

where the last equality follows from the fact that if $\mathbf{B}$ and $\mathbf{C}$ are $m \times n$ and $n \times m$ matrices respectively, then $\det\,(\mathbf{I}_m + \mathbf{BC}) = \det\,(\mathbf{I}_n + \mathbf{CB})$ (e.g.,[11,p.651]). Now, the result follows because $\mathbf{A}^T\mathbf{A} = \mathbf{H}$.

□

It is interesting to particularize this result to the two-user case. Letting $w_i = \int_0^T s_i^2(t)\;dt = H_{ii}$ and ρ equal the normalized crosscorrelation between both signals, i.e., $\rho\sqrt{w_1 w_2} = H_{12} = \int_0^T s_1(t)s_2(t)\;dt$, the capacity region in (2.3) becomes

$$0 \leq R_1 \leq \frac{1}{2}\log\,[1 + \frac{w_1}{\sigma^2}] \tag{2.17a}$$

$$0 \leq R_2 \leq \frac{1}{2}\log\,[1 + \frac{w_2}{\sigma^2}] \tag{2.17b}$$

$$R_1 + R_2 \leq \frac{1}{2}\log\,[1 + \frac{w_1}{\sigma^2} + \frac{w_2}{\sigma^2} + \frac{w_1 w_2}{\sigma^4}\,(1 - \rho^2)] \tag{2.17c}$$

If $|\rho| = 1$, then we obtain the classical Gaussian capacity region [3], in which the sum of the rates depends only on the sum of the signal energies. This succedaneum of Gauss law [2] is no longer valid in our more general multiple-access channel.

A very intuitive and fruitful way to represent the multiple-access capacity region is to consider the *effective* signal-to-noise ratio of each user, $\gamma_i = \exp[2R_i] - 1$, defined as the signal-to-noise ratio required to achieve capacity R_i in single-user communication. Note that the effective signal-to-noise ratio and the rate give the same information. Now it is convenient to define the *efficiency* of each user as the ratio of effective to actual signal-to-noise ratios, i.e.,

$$\eta_i \;=\; \frac{\gamma_i\,\sigma^2}{w_i}. \tag{2.18}$$

The efficiency of each user is a parameter ranging from 0 to 1 that quantifies the degree of performance degradation due to the presence of other active users in the channel. Related measures of degradation due to multiuser interference have proven useful in the analysis of multiuser demodulators for systems with uncoded data [12, 13]. When the efficiency region occupies nearly all of the unit hypercube, then the main mechanism limiting performance is the background Gaussian noise, rather than the multiple-access interference. Conversely, the *asymptotic efficiency* region (as $\sigma \rightarrow 0$) quantifies the underlying limitation of the multiple-access channel due to the crosscorrelation between the assigned signal waveforms.

The symbol-synchronous $K-$user efficiency region can be readily obtained from Proposition 1. Writing the rates in (2.3) in terms of the effective signal-to-noise ratios we obtain

$$\begin{aligned}\sum_{i \in J} R_i &= \frac{1}{2} \log \left[\prod_{i \in J} (1+\gamma_i) \right] \\ &= \frac{1}{2} \log \left[\prod_{i \in J} (1+\sigma^{-2}\, w_i\, \eta_i) \right] \\ &= \frac{1}{2} \log \left[\det \left(\mathbf{I}_{|J|} + \sigma^{-2} \mathbf{W}_J\, \Gamma_J \right) \right] \qquad (2.18)\end{aligned}$$

where $\mathbf{W}_J = diag\{w_i,\, i \in J\}$ and $\Gamma_J = diag\{\eta_i,\, i \in J\}$. Uniting (2.3) and (2.18) we obtain the efficiency region:

$$\bigcap_{J \subset \{1, \cdots K\}} \{\, (\eta_1, \cdots \eta_K) \in [0,1]^K;\ \det(\sigma^2 \mathbf{W}_J^{-1} + \Gamma_J) \le \det(\sigma^2 \mathbf{W}_J^{-1} + \widetilde{\mathbf{H}}_J) \,\}, \qquad (2.19)$$

where $\widetilde{\mathbf{H}}_J$ is the normalized crosscorrelation matrix of the users in J, i.e.,

$$\mathbf{H}_J = \mathbf{W}_J^{1/2}\, \widetilde{\mathbf{H}}_J\, \mathbf{W}_J^{1/2}.$$

It follows from (2.19) that the asymptotic efficiency region is given by

$$\bigcap_{J \subset \{1, \cdots K\}} \{ (\eta_1, \cdots \eta_K) \in [0,1]^K\!:\ \prod_{i \in J} \eta_i \ \le\ \det(\widetilde{\mathbf{H}}_J) \,\}, \qquad (2.20)$$

which, interestingly, is independent of the relative signal energies. Therefore, in order to obtain the asymptotic efficiency region it suffices to compute the determinants of the matrices of normalized crosscorrelations for each subset of users. The expression in (2.20) is especially useful because it gives the number of dB that each group of users have to add to their powers in order to maintain the same rates as in single-user communication. One of the consequences that can be drawn from the region (2.20) is that the maximum number of users with nonzero asymptotic efficiency is equal to the rank of $\mathbf{H}$.

The asymptotic efficiency region is relevant not only because it quantifies the performance degradation in high signal-to-noise ratio channels, but because it is an inner bound to the efficiency region achievable at any noise level. This is a corollary to the following result.

Proposition 2: The multiuser efficiency region (2.19) is monotonically increasing as a function of the noise level σ^2.

Proof: Suppose that the vector $(\eta_1, \cdots \eta_K)$ belongs to the efficiency region achievable at σ_1^2, i.e., for all $J \subset \{1, \cdots K\}$

$$\det(\sigma_1^2 \mathbf{I}_{|J|} + \Gamma_J\, \mathbf{W}_J) \ \le\ \det(\sigma_1^2 \mathbf{I}_{|J|} + \mathbf{H}_J\, \mathbf{W}_J) \qquad (2.21)$$

The goal is to show that (2.21) holds also for $\sigma_2 > \sigma_1$. To that end, recall (e.g., [14,p.70]) that

$$\det(x \mathbf{I}_n + \mathbf{M}) = x^n + \sum_{i=1}^{n} S_i(\mathbf{M})\, x^{n-i} \qquad (2.22)$$

where $S_i(\mathbf{M})$ is the sum of the principal minors of order i of $\mathbf{M}$. Therefore, we can write

$$\det(\sigma_2^2 \mathbf{I}_{|J|} + \Gamma_J\, \mathbf{W}_J) = (\sigma_2^2 - \sigma_1^2)^{|J|} + \sum_{i=1}^{|J|} S_i(\sigma_1^2 \mathbf{I}_{|J|} + \Gamma_J \mathbf{W}_J)\,(\sigma_2^2 - \sigma_1^2)^{|J|-i}$$

$$\det(\sigma_2^2 \mathbf{I}_{|J|} + \mathbf{H}_J\, \mathbf{W}_J) = (\sigma_2^2 - \sigma_1^2)^{|J|} + \sum_{i=1}^{|J|} S_i(\sigma_1^2 \mathbf{I}_{|J|} + \mathbf{H}_J \mathbf{W}_J)\,(\sigma_2^2 - \sigma_1^2)^{|J|-i}.$$

But every principal minor of $\sigma_1^2 \mathbf{I}_{|J|} + \Gamma_J\, \mathbf{W}_J$ [resp. $\sigma_1^2 \mathbf{I}_{|J|} + \mathbf{H}_J\, \mathbf{W}_J$] is equal to $\det(\sigma_1^2 \mathbf{I}_{|Q|} + \Gamma_Q\, \mathbf{W}_Q)$ [resp. $\det(\sigma_1^2 \mathbf{I}_{|Q|} + \mathbf{H}_Q\, \mathbf{W}_Q)$] where $Q \subset J$. Therefore (2.21) implies that

$$S_i(\sigma_1^2 \mathbf{I}_{|J|} + \Gamma_J \, \mathbf{W}_J) \leq S_i(\sigma_1^2 \mathbf{I}_{|J|} + \mathbf{H}_J \, \mathbf{W}_J), \tag{2.23}$$

for all $i = 1, \cdots |J|$ and the proposition follows.

□

In the two-user case the asymptotic efficiency region converges to the intersection of the unit-square with the hyperbolic region $\eta_1 \, \eta_2 \leq 1-\rho^2$. Therefore one can always achieve efficiency equal to $1-\rho^2$ regardless of how close to the single-user transmission rate is the other user's rate. In the conventional scalar multiple-access channel model, it is easy to see that the efficiency region converges to $\{0 \leq \eta_1 \leq 1, \, \eta_2 = 0\} \cup \{\eta_1 = 0, \, 0 \leq \eta_2 \leq 1\}$. as $\sigma \rightarrow 0$. Hence the best strategy is to let only one of the users talk when $|\rho| = 1$ and the signal-to-noise ratio is very high. Note that this is considerably more efficient than Time-Division-Multiple-Access (TDMA) whose asymptotic efficiency is equal to zero for both users — although if *both rates* are required to be nonzero, then TDMA is indeed almost as good as the best coding for low background noise (cf.[15]). Fortunately, none of these conclusions holds in the case where the assigned waveforms are different ($|\rho| < 1$). For example, suppose that $\rho = 0.1$ and two equal-rate equal-energy users have a signal-to-noise ratio of 20 dB and they transmit at the maximum possible rate given by (2.17). Had the users employed TDMA, each of them would have required approximately 40 dB to attain the same rate. But even in the case where there is heavy crosscorrelation between the signals, TDMA is not near-optimum, e.g., if $\rho = 0.9$, then TDMA would still require 33 dB to attain the same rate.

Figure 1 exhibits the effect of a variation of the noise level on the efficiency region. The crosscorrelation between the signals is $\rho = 0.9$; the ratio between the energies of user 1 and user 2 is 5 dB and the signal-to-noise ratio of user 1 varies from -10 dB to 20 dB. The convergence to the low-noise asymptotic efficiency region can be seen for signal-to-noise ratios higher than approximately 15 dB. On the other hand, when the noise is dominant, the efficiency region nearly fills the entire unit square.

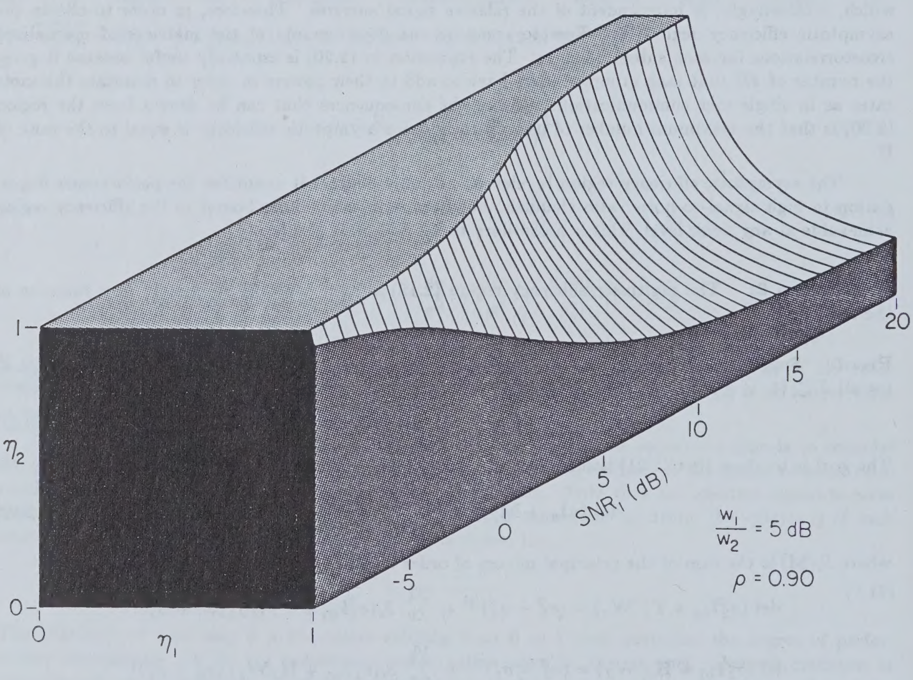


Figure 1. Efficiency region as a function of background Gaussian noise level.

3. Capacity with Amplitude-Constrained Inputs

In this section we investigate the capacity of the amplitude-constrained conventional Gaussian multiple-access channel:

$$Y = \sum_{i=1}^{K} X_i + N \tag{3.1}$$

where the inputs satisfy the constraints $X_i \in [-A_i, A_i]$. This channel has the same capacity as a continuous-time multiple-access white Gaussian channel where all users are assigned the same unit-energy bounded waveform $\{s(t),\ t \in [0,T]\}$:

$$r(t) = \sum_{k=1}^{K} \sum_{i=1}^{n} b_k(i)\, s(t - iT) + n(t) \tag{3.2}$$

and each modulated waveform is restricted to satisfy $\left| \sum_{i=1}^{n} b_k(i)\, s(t - iT) \right| \leq A_k \max_{0 \leq \lambda \leq T} s(\lambda)$. Note that unlike the case where the sufficient statistics of the discrete-time channel are obtained via orthogonal signals with infinite support (e.g. via sampling), in this case input amplitude constraints in the discrete-time channel do translate into amplitude constrained inputs to the continuous-time channel. In the single-user case, Smith [16] showed that the capacity of (3.1) is attained by simple random variables and gave an algorithm for finding the optimum input distribution as a function of the noise level. Rather than generalizing these results to the multiuser case, here we will show a) that if the maximum allowed amplitudes are small enough, then the capacity region is attained by binary inputs and b) the infinite-user capacity region under the assumptions that no user is allowed to dominate and the rate-sum is finite.

Proposition 3: Suppose that X_k is a symmetric random variable with points of increase only on $[-A_k, A_k]$, $k = 1, \cdots K$; and suppose that $(A_1, \cdots A_K)$ is such that $\sum_{i \in J} X_i + N$ is unimodal for all $J \subset \{1, \cdots K\}$. Then the capacity region of (3.1) is equal to

$$\bigcap_{J \subset \{1, \cdots K\}} \{(R_1, \cdots R_K) :\ 0 \leq \sum_{i \in J} R_i \leq h(F^*_{Y_J}) - h(F_N)\} \tag{3.3}$$

where $F^*_{Y_J} = F_N \mathop{*}_{i \in J} F^*_{X_i}$ and $F^*_{X_i}(x) = \frac{1}{2} \quad -A_i \leq x < A_i$.

Proof: The objective is to show that binary equiprobable inputs with maximum allowable amplitude maximize the mutual informations $I(\{X_i\}_{i \in J};\ Y \mid \{X_i\}_{i \notin J})$. Labelling $J = \{a_1, \cdots a_{|J|}\}$ we can write (see [8,p.50])

$$I(\{X_i\}_{i \in J};\ Y \mid \{X_i\}_{i \notin J}) = \sum_{j=1}^{|J|} I(X_{a_j};\ Y \mid \{X_i\}_{i \notin J}, X_{a_1}, \cdots X_{a_{j-1}}). \tag{3.4}$$

Note that the distribution of X_{a_1} only affects the first term in the sum of the right-hand side of (3.4). If we can show that this term is maximized by equiprobable inputs at $\pm A_{a_1}$ regardless of the other input distributions, then the proposition will follow because the ordering of the elements of $J = \{a_1, \cdots a_{|J|}\}$ is arbitrary. In this way, we have converted the problem into one of single-user non-Gaussian capacity. Let F and F_Z be the distribution of X_{a_1} and $Z = \sum_{i=2}^{|J|} X_{a_i} + N$ respectively. Then, the first term in the sum of (3.4) is equal to

$$I(X_{a_1};\ Y \mid \{X_i\}_{i \notin J}) = h(F * F_Z) - h(F_Z). \tag{3.5}$$

Denote the set of symmetric probability distributions with points of increase in $[-A_{a_1}, A_{a_1}]$ by Υ. Then, the problem

$$\max_{F \in \Upsilon} h(F * F_Z) \tag{3.6}$$

is a convex optimization problem for any admissible F_Z (cf. [16]): Υ is a convex set and $h(F * F_Z)$ is concave in $F \in \Upsilon$ (see [17,p.238]).

Therefore, it is enough to show that the directional derivative at $F^*_{X_{a_1}}$:

$$\lim_{\alpha \downarrow 0} \frac{1}{\alpha} \left[h(((1-\alpha) F^*_{X_{a_1}} + \alpha F) * F_Z) - h(F^*_{X_{a_1}} * F_Z) \right] \leq 0, \tag{3.7}$$

for all $F \in \Upsilon$. It is easy to check that the left-hand side of (3.7) is equal to

$$\int_{-A_{a_1}}^{A_{a_1}} f(x) \, (dF(x) - dF^*_{X_{a_1}}(x)) \tag{3.8}$$

where

$$f(x) = - \int_{-\infty}^{\infty} p_Z(y - x) \log \left[\frac{1}{2} p_Z(y - A_{a_1}) + \frac{1}{2} p_Z(y + A_{a_1})\right] dy \tag{3.9}$$

Notice that f is an even function since Z is a symmetric random variable. Hence, in order to show (3.7) it will now suffice to show that f is increasing in the positive real line. But since the density p_Z is differentiable we have

$$f'(x) = \int_0^{\infty} p'_Z(\lambda) \left[g(\lambda + x) - g(\lambda - x)\right] d\lambda \tag{3.10}$$

with

$$g(\lambda) = \log \left[\frac{1}{2} p_Z(\lambda - A_{a_1}) + \frac{1}{2} p_Z(\lambda + A_{a_1})\right]. \tag{3.11}$$

If $x \geq 0$ and $\lambda \geq 0$ then $g(\lambda + x) \leq g(\lambda - x)$ because g is even and decreasing for positive arguments (because $N + \sum_{i \in J} X_i$ is unimodal). Moreover, since Z is also unimodal and symmetric it follows that $p'_Z(\lambda) \leq 0$ for $\lambda \geq 0$. So the integrand in (3.10) is nonnegative, and the proof is complete. □

The set of vectors $(A_1, \cdots A_K)$ that result in unimodal outputs as required in Proposition 3 is an open problem. It can checked that $A_1 \in [0,\sigma]$ and $(A_1,A_2) \in [0,\sigma]^2$ guarantee the unimodality of the output for $K = 1$ and $K = 2$, respectively. Moreover, these are tight estimates in the sense that larger sets $[0,x]^i$ do not guarantee unimodality. Lest one gets too optimistic, $(A_1,A_2,A_3) \in [0,\sigma]^3$ does not guarantee unimodality because of a tiny neighborhood of (σ,σ,σ).

Let us now turn our attention to the asymptotic behavior (as $K \rightarrow \infty$) of the capacity region of the amplitude-constrained Gaussian multiple-access channel. If no user is allowed a dominant amplitude and in the limit the rate-sum is finite, it is necessary that the allowable amplitude of each user goes to zero. As Proposition 3 indicates, capacity is achieved by binary equiprobable inputs with maximum possible variance, and because of the central-limit theorem the sum of any infinite subset of inputs is asymptotically Gaussian. Therefore, in the nontrivial infinite-user case the amplitude constraints are actually equivalent to variance constraints, and the asymptotic capacity region admits an explicit expression. This reasoning is formalized in the following result.

Proposition 4: Consider the Gaussian multiple-access channel

$$Y = \sum_{i=1}^{K} X_i + N$$

where N is $N(0,\sigma^2)$ and X_i is constrained to the interval $[-A_{Ki}, A_{Ki}]$. Suppose that the allowable amplitudes satisfy:

$$\lim_{K \rightarrow \infty} \max_{1 \leq i \leq K} A_{Ki} = 0 \tag{3.12}$$

and

$$\lim_{K \rightarrow \infty} \sum_{i=1}^{K} A_{Ki}^2 < \infty \tag{3.13}$$

If R_{Ki} denotes the rate of the i-th user when the number of users is K, then the limit as $K \rightarrow \infty$ of the capacity region is equal to

$$\bigcap_{J \subset N} \{ 0 \leq \lim_{K \to \infty} \sum_{i \in J_K} R_{Ki} \leq \frac{1}{2} \log \left[1 + \lim_{K \to \infty} \sum_{i \in J_K} \frac{A_{Ki}^2}{\sigma^2} \right] \}$$

where $J_K = J \bigcap \{1, \cdots K\}$.

Proof: Since $var(X_i) \leq A_{Ki}^2$, the $K-$user capacity region is outer-bounded by the conventional Gaussian capacity region [2]

$$\bigcap_{J_K \subset \{1, \cdots K\}} \{ 0 \leq \sum_{i \in J_K} R_{Ki} \leq \frac{1}{2} \log \left[1 + \sum_{i \in J_K} \frac{A_{Ki}^2}{\sigma^2} \right] \} \tag{3.14}$$

Also, the $K-$user capacity region is inner-bounded by the region achievable by a particular choice of the input distributions, namely, $X_i \sim \frac{1}{2}\, \delta_{-A_{Ki}} + \frac{1}{2}\, \delta_{A_{Ki}}$. This region is defined by the mutual informations:

$$I(\{X_i\}_{i \in J_K};\, Y \mid \{X_i\}_{i \notin J_K}) = h(F_{J_K} * F_N) - \frac{1}{2} \log \left[2 \pi e \sigma^2\right] \tag{3.15}$$

where F_{J_K} is the distribution of the binomial random variable $Z_{J_K} = \sum_{i \in J_K} X_i$. The proposition will follow by showing that

$$\lim_{K \to \infty} h(F_{J_K} * F_N) = \frac{1}{2} \log \left[2 \pi e \left(\sigma^2 + \lim_{K \to \infty} \sum_{i \in J_K} A_{Ki}^2\right)\right] \tag{3.16}$$

and hence that the outer bound (3.14) is asymptotically tight. But $N + Z_{J_K}$ is a continuous random variable whose density is over-bounded by $1/\sigma \sqrt{2\pi}$ (because $\frac{1}{2} p(y - A) + \frac{1}{2} p(y + A) \leq \max_z p(z)$); then, the product of that density function and its logarithm is uniformly bounded in K, and therefore the left-hand side of (3.16) is equal to the differential entropy of the limit distribution of $N + Z_{J_K}$. Moreover, since N and Z_{J_K} are independent random variables, it suffices to show that Z_{J_K} converges in distribution (as $K \to \infty$) to $N(0, \lim_{K \to \infty} \sum_{i \in J_K} A_{Ki}^2)$. Since $\sum_{i \in J_K} A_{Ki}^2$ is the variance of Z_{J_K}, the result is true (by L_2-convergence) if $\lim_{K \to \infty} \sum_{i \in J_K} A_{Ki}^2 = 0$. Otherwise, $\sum_{i \in J_K} X_i \,/\, (\sum_{j \in J_K} A_{Kj}^2)^{1/2}$ converges in distribution to a standard Gaussian random variable. In order to show this we verify Lindeberg's condition [18,p. 205]: for each $\delta > 0$,

$$\lim_{K \to \infty} \sum_{i \in J_K} \int_{|x| \leq \delta} x^2 \, dG_{Ki}(x) = 1, \tag{3.17}$$

where G_{ki} is the distribution of $X_i \,/\, (\sum_{j \in J_K} A_{Kj}^2)^{1/2}$ -a binary symmetric random variable with variance $B_{Ki}^2 = A_{Ki}^2 \,/\, (\sum_{j \in J_K} A_{Kj}^2)$. It follows that

$$\sum_{i \in J_K} \int_{|x| \leq \delta} x^2 \, dG_{Ki}(x) = \sum_{i \in J_K} B_{Ki}^2 \, 1\{B_{Ki}^2 \leq \delta^2\}. \tag{3.18}$$

But since $\sum_{i \in J_K} B_{Ki}^2 = 1$, there are always fewer than δ^{-2} terms in J_K such that $B_{Ki}^2 > \delta^2$. Therefore,

$$1 \geq \sum_{i \in J_K} B_{Ki}^2 \, 1\{B_{Ki}^2 \leq \delta^2\} \geq 1 - \delta^{-2} \max_{i \in J_K} B_{Ki}^2 \tag{3.19}$$

and the right-hand side of (3.18) can be made as close to 1 as desired with large enough K because of assumption (3.12).

□

References

1. S. Verdu, "Symbol-Asynchronous Gaussian Multiple-Access Channels," *1986 IEEE International Symposium on Information Theory,* Ann Arbor, Michigan, October, 1986.
2. T. Cover, "Multiple user information theory for the Gaussian channel," *New Concepts in Multi-user Communication,* pp. 53-62, 1981. J.K. Skwirzynski, Ed.
3. A. D. Wyner, "Recent Results in the Shannon Theory," *IEEE Trans. Information Theory,* vol. IT-20, Jan. 1974.
4. A.J. Viterbi, *Spread Spectrum Multiple Access with Binary PSK Modulation can approach Shannon Capacity for the Aggregate Gaussian Channel,* 1986. preprint
5. R.C. Sommer, "Asynchronously Multiplexed Channel Capacity," *Proc. IEEE,* vol. 54, pp. 79-80, Jan. 1966.
6. G. Tartara, "On the Capacity of Asynchronous Code-Division Multiplexing Systems," *Proc. IEEE,* vol. 56, pp. 1357-1358, Aug. 1968.
7. J. Y. N. Hui, "Throughput analysis for Code Division Multiple Accessing of the Spread Spectrum channel," *IEEE Journal Selected Areas in Communications,* vol. SAC-2, pp. 482-486, July 1984.
8. I. Csiszar and J. Korner, *Information Theory: Coding Theorems for Discrete Memoryless Systems,* Academic, New York, 1981.
9. R.J. McEliece, *The Theory of Information and Coding: A mathematical framework for communication,* Addison-Wesley, Reading,Mass, 1977.
10. R. Bellman, *Introduction to Matrix Analysis,* McGraw-Hill, New York, 1960.
11. T. Kailath, *Linear Systems,* Prentice-Hall, Englewood Cliffs, NJ, 1980.
12. S. Verdu, "Minimum Probability of Error for Asynchronous Gaussian Multiple-Access Channels," *IEEE Trans. on Information Theory,* vol. IT-32, pp. 85-96, Jan. 1986.
13. S. Verdu, "Optimum multiuser asymptotic efficiency," *IEEE Trans. Communications,* vol. COM-34, Sept. 1986.
14. F.R. Gantmacher, *The Theory of Matrices,* Chelsea P.C., New York:, 1959.
15. R.G. Gallager, "A Perspective on Multiaccess Channels," *IEEE Trans. Information Theory,* vol. IT-31, pp. 124-142, March 1985.
16. J.G. Smith, "The Information Capacity of Amplitude- and Variance-constrained scalar Gaussian channels," *Information and Control,* vol. 18, pp. 203-219, 1971.
17. R. Ash, *Information Theory,* Wiley Interscience, New York, 1965.
18. K.L. Chung, *A Course in Probability Theory,* Academic Press, New York, 1974. Second Edition

DATA COMPRESSION OF A SIMPLE COMPOSITE SOURCE

JERRY D. GIBSON AND DAE G. JEONG

Department of Electrical Engineering
Texas A&M University
College Station, TX 77843

We consider the data compression of a first-order Laplacian autoregressive process (LAR(1)) given by

$$X_k = \alpha X_{k-1} + \epsilon_k = \begin{cases} \alpha X_{k-1}, & \text{w.p. } \alpha^2 \\ \alpha X_{k-1} + w_k, & \text{w.p. } 1-\alpha^2 \end{cases} \tag{1}$$

where $\{w_k\}$ is an independent and identically distributed sequence of Laplacian random variables with zero mean and $\sigma_w^2 = \sigma_X^2$. The LAR(1) process was first defined by Modestino [1] and has since been studied by several authors [2–5]. The LAR(1) process in Eq. (1) not only holds promise as a model for speech and images, but it is also of theoretical interest since it is a simple example of a composite source as defined by Berger [6]. Composite sources are difficult to handle theoretically; indeed, only bounds on the the rate distortion performance of these sources have been forthcoming [4,7,8]. Additionally, few, if any, studies of the data compression of such sources have been reported.

Here we compare the performance of several different data compression systems for encoding the LAR(1) source. In particular, we examine standard differential pulse code modulation (DPCM), a prediction error coder (PEC)[9], a system proposed for AR sources in 1967 by Davisson [10], and a new coder which attempts to take advantage of the LAR(1) source structure [2]. Since all of these systems are differential encoders involving a predictor and a quantizer, one of the first questions to be answered is how to select the quantizer step size. One common way to choose the step size for a Gaussian AR source is to match it to the variance of the source driving term, which is $\text{var}(\epsilon_k) = (1-\alpha^2)\sigma_w^2$ here. Another popular approach is to use the backward adaptive step size algorithm proposed by Jayant [9,11], where the step size $\Delta(\cdot)$ is adapted based upon the last quantizer output level, viz, $\Delta(k+1) = M(|I_k|)\Delta^\beta(k)$, with $0 < \beta < 1$ and $M(\cdot)$ a multiplier function depending upon the output level at time k. Simulation results with $\alpha = 0.9$ reveal that choosing the step size to be the optimum (minimum mean squared error (MMSE)) step size for a unit variance ($\sigma_w^2 = 1$) Laplacian input outperforms a fixed quantizer matched to $\text{var}[\epsilon_k]$ by 0.4 to 0.55 bits/sample and the Jayant backward adaptive quantizer by more than 1 bit/sample. Thus, classical methods do not work well for this source. The reason for such behavior is the high probability of a zero output level. Although the variance of ϵ_k is $(1-\alpha^2)\sigma_w^2$, the variance of the continuous component is σ_w^2, and the quantizer should be scaled by this latter value to match the nonzero error term. Furthermore, the Jayant backward adaptive quantizer will have too small a step size most of the time because whenever the zero level occurs, the step size contracts. A backward adaptive quantizer which calculates the step size based upon the variance of past coder reconstructed values was also simulated, and it performs about as well as the unit variance, fixed quantizer, but not better.

With the fixed, unit variance, MMSE uniform Laplacian quantizer in the several data compression systems considered, simulation results indicate that DPCM outperforms all of the others, including the system proposed in [2] for the LAR(1) source. Our simulation results are compared to the theoretical results of Farvardin and Modestino [4] when $\alpha = 0.8$ in Fig. 1, which is Fig. 8 of [4] with the simulation data shown by the points labeled "◇". Note that the simulation results fall near their $N = 3$ level, 5 level, and 9 level curves. Theoretical values for quantizers with a larger number of levels are not available from [4]. Note that our simulated performance data seems to be approaching the "Asymptotic Result" derived by Farvardin and Modestino [4].

Other interesting results obtained from the simulations are as follows: (i) The performance of the PEC slowly approaches that of DPCM for the LAR(1) source as the number of quantization levels increases, but for a Gaussian AR(1) source, DPCM has a relatively constant advantage over the PEC of $10\log_{10}(1-\alpha^2)^{-1}$; (ii) for equivalent distortions, the rate required for the LAR(1) source is much less than that for the Gaussian AR(1) source, which is as indicated by the rate distortion curves.

ACKNOWLEDGEMENT

This work was supported, in part, by the Office of Naval Research Statistics and Probability Program under Contract No. N00014-81-K-0299.

REFERENCES

[1] J. W. Modestino, "Nonparametric and Adaptive Detection of Dependent Data," Ph.D. thesis, Electr. Eng. Dept., Princeton Univ., Princeton, NJ, 1969.

[2] J. D. Gibson, "Data Compression of a First-Order Intermittently Excited Process," Proceedings of the Statistical Image Processing and Graphics Workshop, ONR, May 24–27, 1983; also in Statistical Image Processing and Graphics, E. J. Wegman and D. J. DePriest, eds., Marcel Dekker, Inc., 1986.

[3] J. D. Gibson, "The First-Order Laplacian Autoregressive Process," Tech. Rpt., Dept. of Electr. Eng., Texas A&M Univ., Aug. 8, 1984.

[4] N. Farvardin and J. W. Modestino, "Rate-Distortion Performance of DPCM Schemes for Autoregressive Sources," IEEE Trans. Inform. Theory, Vol. IT-31, pp. 402–418, May 1985.

[5] L. S. Dewald and P. A. W. Lewis, "A New Laplace Second-Order Autoregressive Time-Series Model-NLAR(2)," IEEE Trans. Inform. Theory, Vol. IT-31, pp. 645–651, Sept. 1985.

[6] T. Berger, Rate Distortion Theory, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1971.

[7] M. J. Carter and D. L. Neuhoff, "Bounds to the Rate-Distortion Function for Regenerative Composite Sources," Proc. 20th Allerton Conf. Commun., Control and Comput., Monticello, IL, pp. 680–681, Oct. 1982.

[8] M. J. Carter, "Source Coding of Composite Sources," Ph.D. Dissertation, Dept. of Elec. Eng., Univ. of Michigan, Ann Arbor, MI, 1984.

[9] J. D. Gibson, "Adaptive Prediction in Speech Differential Encoding Systems," Proc. IEEE, Vol. 68, pp. 488–525, April 1980.

[10] L. D. Davisson, "An Approximate Theory of Prediction for Data Compression," IEEE Trans. Inform. Theory, Vol. IT-13, pp. 274–278, April 1967.

[11] N. S. Jayant, "Adaptive Quantization With a One-Word Memory," Bell Syst. Tech. J, Vol. 52, pp. 1119–1144, Sept. 1973.

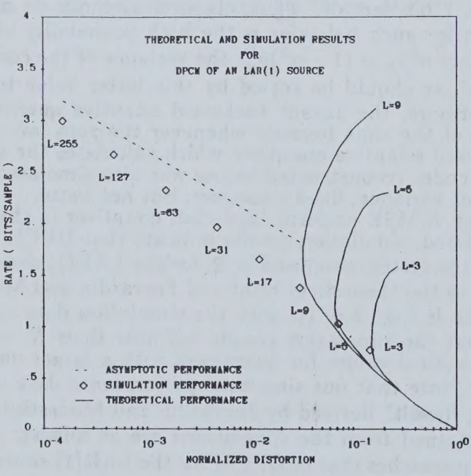


Figure 1.

COMPARISON OF TRELLIS CODED MODULATION SCHEMES USING A LOWER BOUND ON THE FREE EUCLIDEAN DISTANCE

MARC ROUANNE
DANIEL J. COSTELLO, Jr.
Dept. of Elec. & Comp. Engr
Univ. of Notre Dame
Notre Dame, IN 46556

ABSTRACT

A comparison of Trellis Coded Modulation (TCM) schemes is considered. The approach is based upon a lower bound on the minimum free Euclidean distance d_{free} of TCM. The bound is similar to the bound of Costello [1] and Forney [2] on the free distance of convolutional codes. We evaluate the lower bound on d_{free} for various modulation constellations and trellis codes, such as those proposed by Ungerboeck [3], and Lafanechere and Costello [4], and compare it with Calderbank, Mazo, and Wei's upper bound [5,6].

PROBLEM FORMULATION

A TCM scheme is defined by a binary trellis encoder, a signal constellation, and a mapping of signals onto the trellis. Let k be the information block length and ν the memory length of the encoder. Then $\nu_0 = k\,\nu$ is the constraint length of the code (we assume that the encoder contains k memory registers of equal length ν). The encoder is a finite state machine with 2^{ν_0} states and 2^k branches to and from each state. The error probability of a code used with maximum-likelihood (Viterbi) decoding on an AWGN channel can be bounded in terms of its minimum free distance d_{free}. Therefore an efficient mapping will assign channel signals to branches to achieve maximum d_{free} when using maximum likelihood decoding. The lower bound on d_{free} uses a random coding argument based on the moment generating function $\overline{e^{\alpha\rho d^2} \sum_{\underline{y}} \left[\sum_{\underline{y}'} e^{-\alpha d_e^2(\underline{y},\underline{y}')} \right]^{\rho}}$, where the overbar indicates an average over the ensemble of all code

---

This work was supported by NASA grant NAG5-557 and NSF grant ECS84-14608.

choices (for a given trellis), the first sum is over all choices of paths $\underline{y}$, the second sum is over all paths $\underline{y}'$ diverging from $\underline{y}$, α and ρ are arbitrary constants, d is the lower bound on d_{free}, and d_e is the Euclidean distance associated with a particular pair of paths. The lower bound is a function of a trellis and a signal constellation, but not of a specific mapping (i.e., it is a random coding bound). A trellis is characterized by ν_0 and k. A signal constellation is characterized by its dimensionality, energy, point separation and set partitioning. We now see how these parameters affect the lower bound on d_{free}.

DERIVATION OF THE BOUND

Forney [2] set forth clearly the following bounding technique, which owes much to Chernoff [7], Gallager [8], and Viterbi [9]. The technique includes 6 steps. We use only the first 4.

1) Step 1: Define a random ensemble

The random ensemble is the set of all "non-linear, time-varying codes" associated with a specific signal constellation and a specific trellis.

2) Step 2: Obtain a bound on the moment-generating function: $e^{\alpha\rho d^2} \overline{\sum_{\underline{y}} \left[\sum_{\underline{y}'} e^{-\alpha d_e^2(\underline{y},\underline{y}')} \right]^{\rho}}$

If the probability over all codes in the random ensemble that d_{free} is smaller than some d is less than 1, there exists a code for which d_{free} is actually larger than or equal to d. This can be written as follows:

$$\underset{all\ codes}{P}\ (d_{free}(code) < d) < 1 \rightarrow \textit{there exists a code such that } d_{free} \geq d.$$

Furthermore,

$$d_{free}(code) \geq d \quad \text{if} \quad T_{\alpha,\rho}(code,d) < 1$$

where
$$T_{\alpha,\rho}(code,d) \equiv e^{\alpha\rho d^2} \sum_{\underline{y}} \left[\sum_{\underline{y}'} e^{-\alpha d_e^2(\underline{y},\underline{y}')} \right]^{\rho}.$$

Hence,

$$\textit{there exists a code such that } d_{free} \geq d \text{ if } e^{\alpha\rho d^2} \overline{\sum_{\underline{y}} \left[\sum_{\underline{y}'} e^{-\alpha d_e^2(\underline{y},\underline{y}')} \right]^{\rho}} < 1. \qquad (1)$$

3) Step 3: Configuration counting

Equation (1) contains an average over all codes of a function $T_{\alpha,\rho}$. $T_{\alpha,\rho}(code)$ depends on the codewords of a specific *code*. Shannon's random coding technique switches the sum over all codes and the sum over all codewords and obtains an average over all codewords of another function $\hat{T}_{\alpha,\rho}$. $\hat{T}_{\alpha,\rho}(codeword)$ represents the probability over all codes that a specific *codeword* is chosen. Here, the codewords are the paths through the trellis. "Configuration counting" determines the number of occurrences of a specific path in all the codes. This enables us to calculate $\hat{T}_{\alpha,\rho}$.

4) Step 4: Reduce the above bound to a usable form (Gallager's Lemma)

Step 3 puts the bound in random coding form. Step 4 (see Gallager [8]) puts the bound in its final form.

STATEMENT OF THE BOUND

Given a signal constellation S, there exists a (k, ν_0) trellis code with minimum Euclidean distance d_{free} such that:

$$d^2_{free} \geq \max_{\left\{\begin{array}{l} E(\alpha,\rho) > ln2 \\ 0 \leq \alpha \\ 0 \leq \rho \leq 1 \\ p(s) \end{array}\right.} \left[\nu_0 \, \frac{E(\alpha\,,\,\rho)}{\alpha} - \frac{F[\,E(\alpha\,,\,\rho)\,]}{\alpha} \right] \tag{2}$$

where:

k is the information block size,

ν_0 is the constraint length,

S is the signal constellation,

α and ρ are parameters to optimize the bound,

$$E(\alpha,\rho) = -\ln\left[\sum_{s \in S} p(s) \left[\sum_{s' \in S} p(s')\, e^{-\alpha\, d_e^2(s\,,\,s')} \right]^{\rho} \right]^{\frac{1}{\rho k}} ,$$

s and s' are signal points from the signal constellation S,

$d_e(s,s')$ is the Euclidean distance between the signal points s and s',

and $p(s)$ is a probability distribution on S.

APPLICATION OF THE BOUND

A signal s can itself be a subset of the constellation when a trellis contains parallel transitions. The subset is composed of signals assigned to the parallel transitions. This enables us to study the effects of uncoded bits in the encoder, or, equivalently, the effects of mapping signals onto a trellis with parallel transitions using Ungerboeck's "set partitioning" method [3].

A trellis is entirely characterized by k and ν_0. Thus each evaluation of the bound corresponds to one trellis, and represents the average free distance of all the corresponding trellis codes. However, since the bound is a random coding bound, it does not represent the best free distance achievable with a particular trellis.

Lower bounds on d_{free} have been obtained for binary convolutional codes using the Hamming distance metric [1,2]. Like the lower bounds on the Hamming d_{free}, this lower bound on the Euclidean d_{free} is exponentially tight and varies linearly with ν_0 for large ν_0. Like any other "Gilbert type" bound, this lower bound is not expected to be tight for small ν_0.

The bound depends explicitly on the constraint length ν_0 and implicitly on the information rate per dimension $R \equiv \frac{k}{dim}$. Although only certain discrete values of ν_0 and R are possible, we sketch $d^2_{free}(\nu_0)$ and $\frac{d_{free}}{\nu_0}(R)$ as continuous functions.

USING THE BOUND TO EVALUATE TCM SCHEMES

1) d_{free} viewed as a function of ν_0:

The lower bound on d_{free} increases with the constraint length ν_0. Bounds on the free Hamming distance of binary convolutional codes have usually been expressed as functions of ν_0. This leads to asymptotic, exponentially tight bounds on d_{free} when ν_0 is large but the bounds are not tight for small constraint lengths. In the following paragraphs we compare constellations at a given rate R of information bits per dimension and at a given average energy E_{avg} per dimension.

* The effect of increasing the number of signals on d_{free} (same rate R, same dimensionality dim, same average energy E_{avg}):

Increasing the number M of signals increases d_{free} because it provides more flexibility in the choice of signals assigned to trellis branches (flexibility in the mapping). Signals with greater distance can be chosen on branches of merging paths, which increases d_{free}. Note that increasing M, while keeping k constant, lowers the code rate $R_c \equiv \frac{k}{\log_2 M}$. Fig.1 shows PAM schemes with different numbers of signals ($M = 4, 8, 16, 128$). Increasing the number of signals provides more gain for small constraint lengths ν_0 than for large constraint lengths, because when ν_0 is large flexibility in mapping is provided by the complexity of the encoder. In Fig.1, going from 4-PAM to 8-PAM or even 16-PAM provides most of the gain and higher order PAM schemes do not improve d_{free} significantly. Note that the gain in d_{free} may not be worth the added modulation complexity.

* The effect of increasing the signal set dimensionality on d_{free} (same rate R, same number of signals M, same average energy E_{avg}):

Increasing the signal set dimensionality increases d_{free}. Higher dimensional schemes are obtained by using a basic one- or two-dimensional constellation for several transmission intervals [4]. Fig.2 shows L-dimensional schemes for which the constellation is constructed from L uses of a given 1-dimensional constellation (here 4-PAM). For a given constraint length ν_0, higher dimensionality yields a larger d_{free}, and as ν_0 increases the curves diverge and the gain goes to infinity. However, the error coefficient increases with dimensionality [10], which may cancel the gain from the increased d_{free} at moderate decoded bit error rates.

* The effect of changing the modulation scheme on d_{free} (same dimensionality dim, same number of signals M, same average energy E_{avg}, same rate R):

Changing the modulation scheme affects d_{free}. For example, rectangular constellations yield a larger d_{free} than constant envelope constellations. Fig.3 shows M-PSK vs. M-QASK. This suggests that a d_{free} penalty is associated with constant envelope modulation. Note that this penalty increases with the constraint length. Furthermore, the bound gives a means of optimizing some parameters associated with a specific type of constellation without searching exhaustively for the best constellation.

* The effect of uncoded bits or parallel transitions on d_{free} when the signal constellation is "mapped by set partitioning":

Uncoded bits generate parallel transitions in the trellis and limit d_{free}, since d_{free} cannot be larger than the minimum distance between parallel transitions (the minimum distance within the subsets created by "set partitioning"). Fig.4 shows that for 8-PSK, 1 uncoded bit increases d_{free} for small constraint lengths. For small ν_0, this agrees with the best known codes constructed [3]. For large ν_0, however, we see that 1 uncoded bit limits the achievable d_{free}. The bound shows whether uncoded bits combined with mapping by set partitioning will increase d_{free} for small constraint lengths ν_0.

2) Comparison of the lower bound on d_{free} with upper bound [5,6] and known codes [3]:

The upper bound is expected to be tighter than the lower bound for small constraint lengths, and the lower bound is expected to be tighter for large constraint lengths (this is analogous to the way bounds on binary convolutional codes behave). The lower bound is a "Chernoff type" bound and therefore is exponentially tight when ν_0 is large. Fig.5 shows that the upper bound is tighter than the lower bound for small constraint lengths. But the lower bound becomes tighter as ν_0 increases. The slope of the lower bound gives a precise indication of the asymptotic rate of increase in d_{free}. Finally, the lower bound guaranties the existence of codes that can achieve a certain d_{free}.

3) Asymptotic behaviour: $\frac{d_{free}}{\nu_0}$ as a function of $R = \frac{k}{dim}$ for large ν_0.

For a given constellation, the achievable $\frac{d_{free}}{\nu_0}$ is larger for small R than for large R. Graphs of $\frac{d_{free}}{\nu_0}(R)$ give a means of comparing the asymptotic behaviour (large ν_0) of d_{free} for different constellations. Several examples are shown in Fig.6. The largest achievable rate R depends on the constellation and its dimensionality : it is $\frac{\log_2 M}{dim} R_c$ for a code rate R_c. These largest achievable rates appear clearly in Fig.6. Rectangular lattices constructed from the same 1-dimensional constellation (for example 4-PAM in Fig.6) yield the same graph, which means that these rectangular lattices have the same asymptotic behaviour. In other words the curves $d_{free}(\nu_0)$ are parallel for large ν_0. The lower bounds are compared with an asymptotic upper bound [6] in Fig.6.

CONCLUSION

We have obtained a random coding lower bound on d_{free}. It takes into account the actual Euclidean distances between points of the channel signal constellation. As with any "Gilbert" bound, it is most useful as an asymptotic bound. It provides a means of comparing the asymptotic performance of different modulation schemes.

REFERENCES

[1] D. J. Costello, Jr. "Free Distance Bounds for Convolutional Codes", IEEE Trans. Inform. Theory, vol. IT-20, pp. 356-365, May 1974.

[2] G. D. Forney, Jr. "Convolutional Codes II: Maximum Likelihood Decoding", Information and Control , vol. 25, pp. 222-266, July 1974.

[3] G. Ungerboeck, "Channel Coding with Multilevel Phase Signals", IEEE Trans. Inform. Theory, vol. IT-28, pp. 55-67, Jan. 1982.

[4] A. Lafanechere and D. J. Costello, Jr., "Multidimensional Coded PSK Systems Using Unit-Memory Trellis Codes", Proceedings Allerton Conf. on Communication, Control, and Computing, pp. 428-429, Monticello, IL, Sept. 1985.

[5] A. R. Calderbank, J. E. Mazo, and V. K. Wei, "Asymptotic Upper Bounds on the Minimum Distance of Trellis Codes", IEEE Trans. Commun. , vol. COM 33, no. 4, pp. 305-310, April 1985.

[6] A. R. Calderbank, J. E. Mazo, and H. M. Shapiro, "Upper Bounds on the Minimum Distance of Trellis Codes", Bell Syst. Tech. J. vol. 62, pp. 2617-2646, Oct. 1983, part I.

[7] H. Chernoff, "A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on a Sum of Observations," Ann. Math. Stat. , vol. 23, pp. 493-507, 1952.

[8] R. G. Gallager, "A simple Derivation of the Coding Theorem and Some Applications," IEEE Trans. Inform. Theory , vol. IT-11, pp. 3-18, Jan. 1965.

[9] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," IEEE Trans. Inform. Theory , vol. IT-13, pp. 260-269, Apr. 1967.

[10] G. D. Forney, Jr., R. G. Gallager, G. R. Lang, F. M. Longstaff, and S. U. Qureshi, "Efficient Modulation for Band-limited Channels", IEEE J. on Selected Areas in Communication , vol. SAC-2, no. 5, pp. 632-647, Sept. 1984.

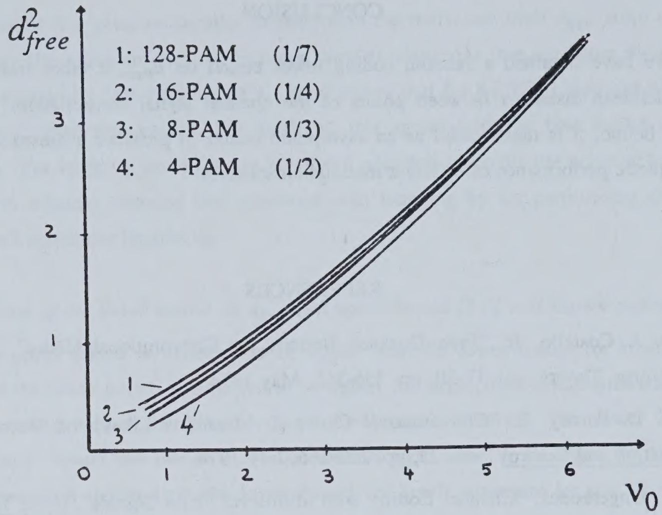


Fig. 1: PAM comparison, $R = 1 bit/dim$, $E_{avg} = 1$.

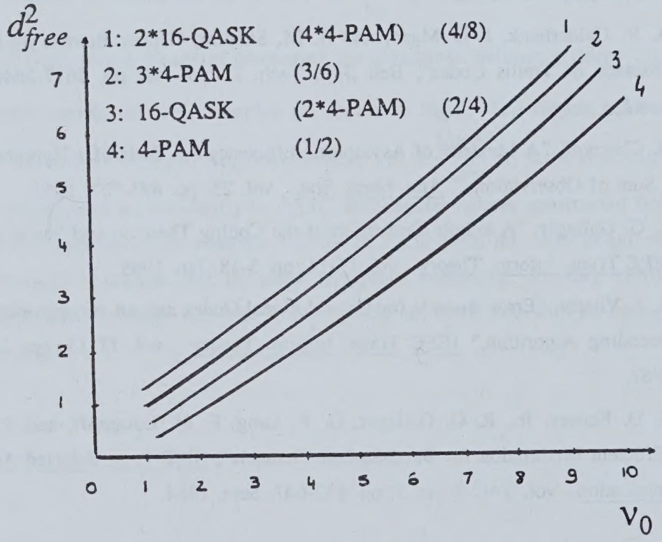


Fig. 2: Comparison of schemes with different dimensionalities, $R = 1 bit/dim$, $E_{avg} = 1$.

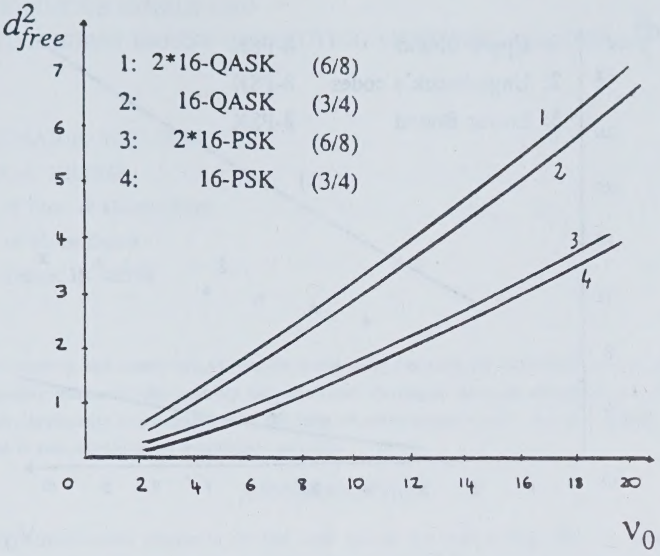


Fig. 3: Comparison of M-PSK, M-QASK, L*M-PSK and L*M-QASK schemes, $R = 1.5$ *bit/dim*, $E_{avg} = 1$.

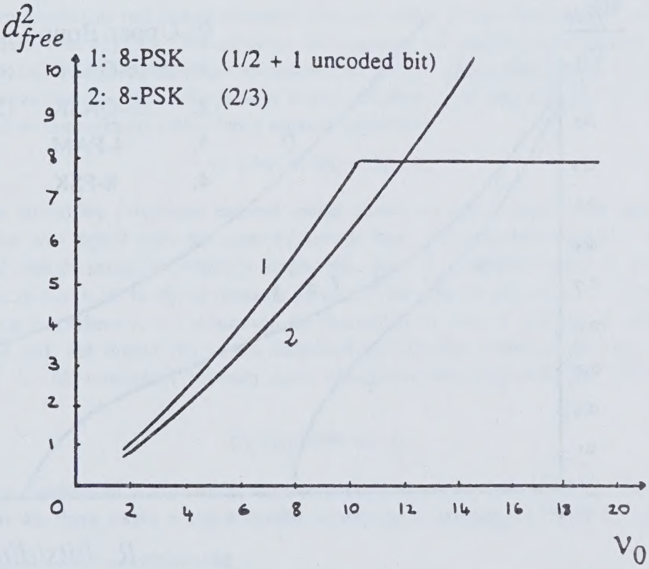


Fig. 4: Trellis coding and uncoded bits (Ungerboeck) for 8-PSK. $R = 1$ *bit/dim*, $E_{avg} = 1$.

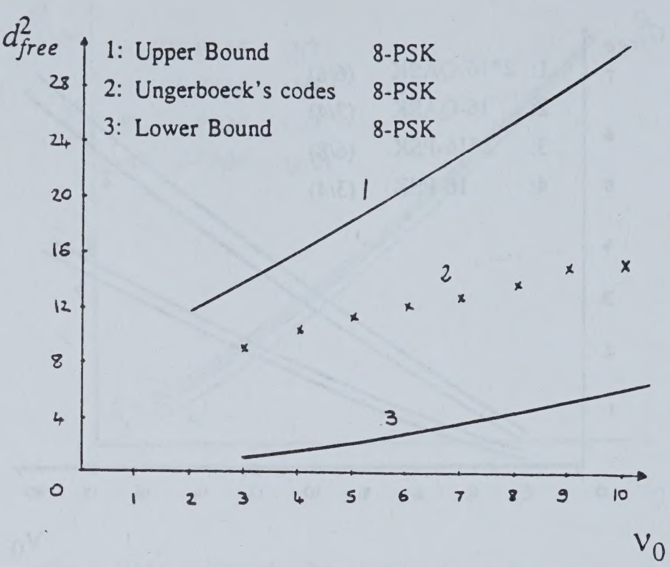


Fig. 5: Comparison of the lower bound on d^2_{free} with upper bounds (Calderbank, Mazo, and Wei).

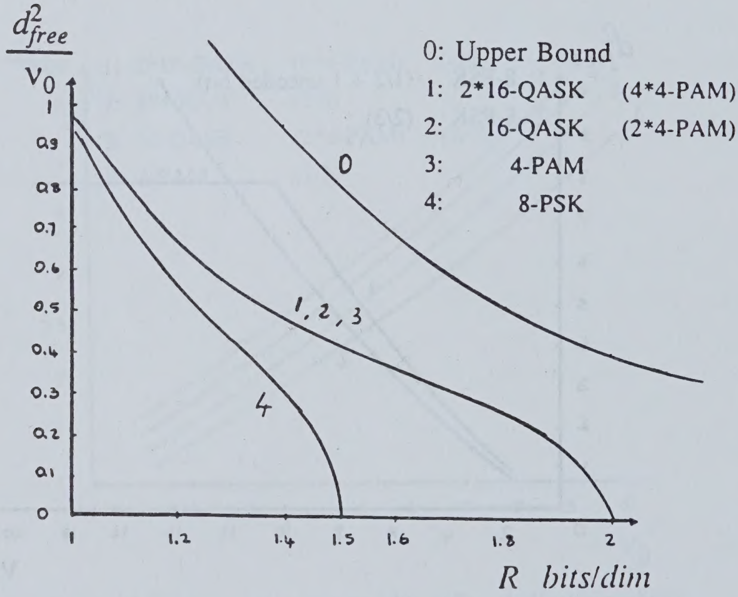


Fig. 6: Asymptotic bounds vs rate, $E_{avg} = 1$.

RANDOM BURST CHANNELS AND AN ADAPTIVE BURST ERROR CORRECTING VITERBI ALGORITHM

CHRISTIAN B. SCHLEGEL
MARK A. HERRO
Dept. of Elec. & Comp. Engr.
Univ. of Notre Dame
Notre Dame, IN 46556

Abstract: The capacity and cutoff-rate of random burst noise channels are calculated. It is shown numerically that while capacity increases with memory the cutoff-rate decreases. As both parameters are channel quality measures, their significance is compared with the help of performance results of a new adaptive Viterbi algorithm designed to take advantage of a channel's memory.

I. INTRODUCTION

Communication channels in the real world are often degraded by noise of a time dependent nature, also known as nonuniform noise. Most information theoretical analysis though primarily deals with memoryless channels, and few solid results on channels with memory and efficient coding for them has come forth. In addition, a great number of descriptive and generative mathematical models for the nonuniform channel have been proposed [1], most of which either lack the mathematical simplicity to be applied to pending coding questions or are not accurate enough to describe the memory process satisfactorily. In this paper we shall use the partitioned Markov chain model first proposed by Fritchman [2] in all mathematical analysis. This model has been applied to real communication channels where it has shown good agreement with experimental data. (see [2],[3]). We calculate the capacity and cutoff-rate with and without side information for this channel and deduce an opinion on which coding strategies are most promising in this burst environment. We will present a new adaptive decoding scheme which employs a Viterbi decoder in conjunction with a burst sensing algorithm.

II. CHANNEL MODEL

The simplified Fritchman channel model which we use is extensively described in [2] and [3] and we will repeat only the essential details here. The model consists of a finite state Markov chain with N states, of which a single one, state N, produces errors in the transmitted channel bits x_i, in that a '1' is added (modulo 2) to x_i if the channel is in state N at the time interval i, i.e., $y_i = x_i + e_i$ where $e_i = 1$ whenever the channel is in state N, and $e_i = 0$ otherwise. The channel model and the matrix (P) of the transition probabilities between the channel states are shown in Fig. 1. All transitions between states occur synchronously with the channel transmissions.

III. CUTOFF RATE

One measure of a channel's transmission quality is the cutoff-rate R_0. It is the largest number for which there exists a linear bound on the error probability P_B of a "good" code of

---

This work was supported by NSF grant ECS84-14608.

length N and rate R, i.e $P_B \le 2^{-N(R_0-R)}$. Furthermore, it is widely believed to be the largest rate for which practical coding systems are feasible (e.g., McEliece & Stark [4] and Massey [6]).

A. Cutoff Rate without Side Information

The channel cutoff-rate is calculated by considering the error probability, P_B, of random codes with two code words and averaging this probability with respect to all possible codes, i.e,

$$\overline{P_B} \le \sum_{x,x'} Q(x)Q(x')\sum_{y}\sqrt{p_N(y|x)p_N(y|x')}, \tag{1}$$

where Q(x) and Q(x') are the probabilities of choosing the code words x and x' randomly and independently from each other. The average block error probability may be upper bounded (see Viterbi [9] or Massey [7]) by

$$\overline{P_B} \le \sum_{y}\left\{\sum_{x} Q(x)\sqrt{p_N(y|x)}\right\}^2 \tag{2}$$

where y is the received sequence of length N. As $p_N(y|x)$ is just the probability that a particular error pattern occurs, we obtain

$$\overline{P_B} \le \sum_{y}\left\{\sum_{e_N} Q(y-e_N)\sqrt{p_N(e_N)}\right\}^2 \tag{3}$$

where e_N is the error pattern from x to y. A detailed look at the channel, with the input alphabet x and output alphabet y, reveals that it is a uniformly focusing and uniformly dispersive channel as discussed in Massey [7]. This is due to the independence of the error from the transmitted sequence. The channel is therefore symmetric and the minimizing distribution in (3) is the uniform one, i.e $Q(x) = \frac{1}{2^N}$, yielding

$$\overline{P_B} \le \sum_{y}\left\{\sum_{e_N} \frac{1}{2^N}\sqrt{p_N(e_N)}\right\}^2. \tag{4}$$

In the following we assume that n-bit-block interleaving is used on the channel, i.e. the channel is partitioned into blocks of n bits and these blocks are spaced g time units apart, where g is chosen large enough to insure independence between the various blocks. We now may rewrite $p_N(e_N)$ as $\pi_{i=1}^{N/n}\, p_n(e_{ni})$, where e_{ni} is a block error pattern of length n, and n divides N. We then may rewrite (4) as

$$\overline{P_B} \le \sum_{y}\frac{1}{2^{2N}}\left\{\sum_{e_{n1}}\cdots\sum_{e_{n\frac{N}{n}}}\pi_{i=1}^{N/n}\sqrt{p_n(e_{ni})}\right\}^2 = \sum_{y}\frac{1}{2^{2N}}\left\{\pi_{i=1}^{N/n}\sum_{e_{ni}}\sqrt{p_n(e_{ni})}\right\}^2 \tag{5}$$

The summation over y above yields the same term 2^N times and, noticing that e_{ni} is merely a dummy variable, we may drop the subscript, and obtain

$$\overline{P_B} \le \frac{1}{2^N}\left\{\sum_{e_n}\sqrt{p_n(e_n)}\right\}^{\frac{2N}{n}} = 2^{-N\left[-\log_2\frac{1}{2}\left\{\sum_{e_n}\sqrt{p_n(e_n)}\right\}^{\frac{2}{n}}\right]} \tag{6}$$

where the summation now is over all the possible error patterns e_n of length n. So we have the desired form of the bound

$$\overline{P_B} \le 2^{-NR_0(n)} \quad \text{where} \quad R_0(n) = 1 - \frac{2}{n}\log_2\sum_{e_n}\sqrt{p_n(e_n)}, \tag{7}$$

and $p_n(e_n)$ is the probability of the error pattern e in a sequence of length n. This rate is a function of n, where n has the significance of a channel memory length.

Note that $1/n$ is proportional to the decoding delay introduced by the block interleaving i.e., if n=1 interleaving (bit interleaving) requires a total delay of D=Ng, n-block interleaving in general requires $D = Ng/n$. In this sense the cutoff-rate curve provides a relationship between decoding delay and achievable error performance.

The mathematical form of this result and its numerical behaviour as a function of n is similar to that found by McEliece and Stark in [4] for a Block Interference channel.

B. Cutoff Rate with Side Information

Let us define what we term side information. We assume that the receiver is somehow able to attach a reliability measure to the received channel bits. Obviously if we knew in which state the channel was at any given transmission interval we would be able to retrieve the information without errors. We will assume that our receiver is, however, equipped to distinguish between two types of states, which we call the burst states and the guard states. The burst states comprise the k noisiest channel states, i.e, the error state N and the k-1 states with the highest transition probabilities into the error state. The N-k remaining states make up the noiseless guard states as shown in Fig. 2. Whenever side information is available we will refer to the channel being in the noisy state by $s_i = B$, and $s_i = G$ if the channel is in the noiseless state. The choice of k seems at first arbitrary but in the actual numerical model, the identification of k is pretty obvious. The random coding bound then takes the form

$$\overline{P_B} \leq \sum_{x,x'} Q(x) Q(x') \sum_{y} \sum_{s} \sqrt{p_N(y,|x) p_N(y,s|x')} \,, \tag{8}$$

where s is the state vector of length N. In a similar approach to that used in section III(A) we arrive at the following result.

$$\overline{P_B} \leq 2^{-NR_0(n)} = \left[\sum_{s_n} P(s_n)\, 2^{-NR_0(n,s_n)} \right]^{N/n} , \tag{9}$$

$$\text{where} \quad R_0(n,s_n) = 1 - \frac{2}{n} \log_2 \sum_{e_n} \sqrt{p_n(e_n|s_n)} \quad \text{and} \quad R_0(n) = \frac{1}{N} \log_2 \left[\sum_{s_n} P(s_n)\, 2^{-NR_0(n,s_n)} \right]^{N/n} . \tag{10}$$

IV. CHANNEL CAPACITY

Although the channel cutoff-rate suggests an upper bound on coding rate for practical codes, Shannon's capacity provides the fundamental limit on transmission rate. Channel capacity is considered as one of the most, if not the most, single important parameter characterising noisy channels.

A. Capacity without Side Information

Fritchman [2] has shown that the capacity C for his Markov chain channel without side information is

$$C = 1 - \lim_{p \to \infty} h(e_p|e_1 \cdots e_{p-1}) \,, \tag{11}$$

where $h(e_p|e_1 \cdots e_{p-1})$ in (1) and (2) is the conditional entropy of the p–$\underline{th}$ error symbol, which can be written as

$$h(e_p|e_1 \cdots e_{p-1}) = - \sum_{m=0}^{p-2} P(10^m) \sum_{e_p} P(e_p|10^m) \log_2 P(e_p|10^m) \tag{12}$$

$$- P(0^{p-1}) \sum_{e_p} P(e_p|0^{p-1}) \log_2 P(e_p|0^{p-1}) ,$$

where 10^m denotes m zeros following an error, and 0^{p-1} is a string of p-1 zeros. These quantities can easily be evaluated from the channel model parameters (see for example Drukarev and Yiu [3]). The actual channel we are discussing here consists of the combination of the encoder, the physical channel and the decoder. The user of this 'super' channel customizes his channel by grouping the input stream into blocks of length *n*, and we may calculate the capacity for the resulting 2^n-ary Discrete Memoryless Channel (DMC). This approach is in keeping with the method used to compute the cutoff-rate. According to Viterbi [9] the capacity of a DMC may be evaluated as

$$C_{DMC} = \sum_{y_n} \sum_{x_n} p(y_n|x_n) q(x_n) \log_2 \frac{p(y_n|x_n)}{p(y_n)}$$

$$= \sum_{e_n} p(e_n) \log_2 \left[p(e_n)\, 2^n \right]$$

$$= n + \sum_{e_n} p(e_n) \log_2 p(e_n), \tag{13}$$

which may be expressed on a capacity per bit basis as

$$C = 1 + \frac{1}{n} \sum_{e_n} p(e_n) \log_2 p(e_n). \tag{14}$$

Equations (11) and (14) yield the same result in the limit as n and p go to infinity, i.e.,

$$1 - \lim_{p\to\infty} h(e_p|e_1 \cdots e_{p-1}) = 1 + \lim_{n\to\infty} \frac{1}{n} \sum_{e_n} p(e_n) \log_2 p(e_n) . \tag{15}$$

B. Capacity with Side Information

If we are in possession of side information, that is if we know the channel state sequence $\{s_i\}$, capacity can be calculated as

$$\overline{C} = 1 - \lim_{p\to\infty} h(e_p|e_1 \cdots e_{p-1}, s_1 \cdots s_p) \tag{16}$$

where the s_i denote the state type, that is $s_i \,\varepsilon\, \{B,G\}$, where B denotes the bursty states and G denotes the guard states. We may express the conditional entropy of the *p–th* channel bit, given the p-1 previously received channel bits and the state type sequence $\{s_i \cdots s_p\}$ as

$$h(e_p|e_1 \cdots e_{p-1}, s_1 \cdots s_p) = \tag{17}$$

$$P(1) \sum_{m=0}^{p-2} P(0^m|1, s_{p-1})\, h\left[\frac{P(0^{m+1}|1, s_{p-1}, s_p)}{P(0^m|1, s_{p-1})} \right] + P(0^{p-1}|, s_{p-1}, s_p)\, h\left[P(0|0^{p-1}, s_{p-1}, s_p) \right]$$

where we have used $s_p{=}B$ and $s_{p-1}{=}B$ wherever these states are referenced.

V. NUMERICAL EVALUATION

For a numerical evaluation of the above parameters we chose the channel reported by Drukarev and Yiu in [3]. This particular channel seemed a good starting point as it exhibits very dense bursts that occur in long clusters. Drukarev and Yiu have extensively tested the performance of various coding schemes on this channel and have reported that the method of interleaving random error correcting codes yielded by far the best results, which is not surprising in light of the results we have found for the cutoff-rate parameter for this channel (Fig. 3). Interleaving random error correcting codes to combat burst noise is a popular technique and can be applied to any bursty channel. In general, this technique is robust in that changes in the channel statistics do not influence performance appreciably.

The value of R_0 decreases with growing n while capacity increases slightly (see Fig. 4.). The same phenomenon has been observed for other channels with memory [4]. We arrive at the crossroads where we have to decide which parameter characterises a channel's performance more accurately, as both R_0 and C are believed to be channel quality measures. This decision is of more than academic interest as it will shed light on the question of whether or not interleaving is in general the superior coding approach on channels with memory.

The steep increase in capacity with the first few observed channel bits further suggests that a decoder memory of ten or so channel bits will suffice to make use of the channel's increased capacity and that a longer memory will be of little benefit. The adaptive schemes discussed in the next section operate precisely on this basis.

The application of single burst error correcting codes on these channels yielded rather discouraging results. Fig. 5 shows the error performance of some single error-correcting codes, and in general a performance degradation rather than improvement was observed with all single burst-error-correcting codes, on dense as well as on diffuse burst noise channels.

VII. ADAPTIVE SCHEME

Giving more credence to the capacity measure, we believe that a direct approach to the burst property of these channels should give better or comparable results than the interleaved codes. Adaptive coding schemes have the advantage that they can deal with bursts of almost any length and do not, like burst error correcting block codes, depend on a 'clean', i.e. error free, guard space immediately following that burst. They observe a short string of channel bits and decide if the channel is in a burst or not. According to the capacity curves observing only a few channel bits at a time is sufficient to make use of the channel's memory.

The basic philosophy of all adaptive schemes is that the information bits appear at least twice in the encoded sequence. Once they are sent in unaltered form in a systematic code, and once they are sent superimposed on later parity bits, along with other information bits. If a burst is detected, i.e., the decoder determines that it cannot correctly decode a given received sequence, the decoder enters its second mode of operation, the burst mode. In this mode the information bits are retrieved from the later parity bits on which they have been superimposed during the encoding operation. The sequence during which this recovery operation happens is called the recovery area. By sending parity bits and delayed information bits in the same channel symbol, the original rate of the code is preserved.

The generally very good performance of adaptive schemes, (see e.g., Massey [8]), on dense burst noise channels can be understood by examining the capacity curves for these channels more closely. Fig. 4 show that almost all of the capacity gain due to memory lies within about 20 observed channel bits. The implication is that capacity can be achieved by basing the decision about an individual bit on the no more than 20 previously received bits. In an adaptive scheme 20 is roughly the number of channel bits the decoder needs to determine whether or not the channel is in a burst. This is a strong argument in favour of capacity as the superior channel quality measure.

VIII. ADAPTIVE VITERBI ALGORITHM

Some of the limitations of adaptive schemes, like the Gallager burst finding scheme (see for example Costello [5]), are that it can use only systematic codes with threshold decoding which is, while easily implementable, a suboptimal decoding method. Furthermore, the Gallager scheme bases its burst decision on a short sequence of observed symbols only, which is a poor match to some diffuse burst noise channels. For those channels we need to take more successive bits into account for our decision. The Adaptive Viterbi Algorithm is designed to overcome these shortcomings. It too, operates in two modes (random and burst mode) and allows a maximum number of errors t, to be corrected in a received sequence of length l branches. The numbers t and l are variable and can be matched to a particular channel. In order to make use of the code's error correcting power however, we suggest choosing $t \geq \lfloor (d_{free}-1)/2 \rfloor$, where d_{free} is the code's free distance. If the number of errors exceeds t in any contiguous sequence of l received bits, the decoder will declare a decoding failure and switch to the burst mode.

The parameter l has to be chosen so that the leading edge of a burst is detected before the burst has moved l time units into the decoder. It can be shown that the probability of not detecting the burst can be made arbitrarily small. A good value of l is 3 to 4 times the constraint length of the code.

The choice of the parameter L will depend on the particular channel on which the system is used. L has to be large enough to include all but a small fraction of the bursts. Since a burst that is longer than L will ultimately cause decoding errors, this small fraction has to be significantly smaller than the system's target error probability. The choice of L can easily be determined from the channel statistics. With hard decision decoding, the metrics used in the Viterbi algorithm are simply the Hamming distances between the received sequence and the respective surviving paths. The method can be generalized to include soft decision decoding by making t a real valued threshold.

Let us now examine the proposed algorithm in more detail. Figure 6 shows the encoder of a rate 1/2 convolutional code. Note that the information bits x_i are delayed an additional L time units after which they are added modulo 2 to the second code bits, i.e.

$$c_i^{(2)} = x_{i-L-m} + v_i^{(2)}. \tag{18}$$

where $c_i^{(2)}$ and $c_i^{(1)}$ are the bits that are actually sent over the channel at time unit i, and m is the encoder memory order. At the decoder (Fig.7) the inverse operation takes place and the Viterbi decoder operates on the received bits $v_i^{(2)}$ and $v_i^{(1)}$. The delay d, (the truncation length), as part of the total delay L, is introduced by the Viterbi decoder. The adaptive decoder looks not only at the metrics of each survivor at time unit i but also at their metrics l branches earlier. For each surviving path s, the decoder considers the metrics $m_{i,s}$ and $m_{i-l,s}$, where $m_{i-l,s}$ is the metric of this survivor l time units earlier. At each decoding step i the difference $m_{i,s} - m_{i-l,s}$ is monitored for each survivor and if

$$m_{i,s} - m_{i-l,s} \leq t \tag{19}$$

for at least one of them, random mode decoding continues; otherwise a burst alarm is sounded. Violation of condition (19) for all surviving paths means that all possible paths through the trellis disagree with the received sequence in at least t positions over a length of l received bits. This is understood as the detection of a burst. Assuming that condition (19) is violated at time unit k for the first time, the decoder will declare the start of a burst at position k–l. and cancel all the decoded information bits until condition (19) holds again for at least one of the survivors during M successive decoding steps. In Fig. 8 this is assumed to have happened at position j+M and a total of j–k+M+l information bits have therefore been cancelled. During this whole operation Viterbi-decoding continues internally.

The cancelled section of these information bits is retrieved L+m time units later during the error free guard space. The decoder then switches from Viterbi-decoding to burst decoding which operates in the following fashion. (see Fig. 7 and Fig. 8).

At time unit $k-l+L+m$ the survivor with the lowest metric is assumed to accurately reflect the transmitted sequence and control is turned over to the burst decoder, which for the next $l+M+(j-k)$ decoding steps (i.e. the cancelled section) will provide the output bits $(x_{k-l}^{[B]} \cdots x_{j+M}^{[B]})$, where $x_i^{[B]}$ refers to the information bits recovered during burst mode operation. The burst decoder is an algorithm that simply walks through the code trellis, assuming the incoming received bits are correct! This walk starts out in the state associated with the survivor with the lowest metric at time unit $k-l+L+m$. The first received bit per branch, $v_i^{(1)} = c_i^{(1)}$ determines the extension of the 'correct' survivor. This extension will always be unequivocal for good codes and will coincide with the transmitted sequence as long as we are in the error free guard space. The second received bit per branch, $c_i^{(2)}$, is added modulo 2 to the second bit found on the code trellis branches of the extended survivor as we walk through the trellis. This modulo 2 addition yields the delayed encoded information bit x_{i-L-m}. The decoder remains in this burst decoding mode for $M+l+(j-k)$ time units. As the trellis walker supplies the second branch bits $v_i^{(2)}$ to the Viterbi decoder for the duration of the burst, it will find exactly one path through the trellis. The metric of this path will normalize itself to zero, and at time unit $j+M+L+m$, when Viterbi-decoding resumes, the decoder will start out in the correct state, provided the channel bits $[c_{k-l+L+m}^{(1)} c_{k-l+L+m}^{(2)} \cdots c_{j+M+L+m}^{(1)} c_{k-l+L+m}^{(2)}]$ came from the error free guard space.

X. CONCLUDING DISCUSSION

The capacity curves suggest that it is possible to achieve good error performance on channels with memory without necessarily resorting to interleaving. The random coding bound on error performance for these channels tell us that we must accept a certain increase in the overall decoding delay in order to achieve good performance (derived for block interleaving). The performance of the suggested Adaptive Viterbi Algorithm implies that the significance of $R_0(n)$ as an inverse measure of decoding delay is not confined to the block interleaved case but is general in nature.

REFERENCES

[1] L. N. Kanal and A. R. K. Sastry, "Models for Channels with Memory and Their Applications to Error Control", *Proceedings of the IEEE,* vol. 66, July 1978, pp. 724-743.

[2] B. D. Fritchman, "A Binary Channel Characterization using Partitioned Markov Chains", *IEEE Trans. Inform. Theory,* vol IT-13, April 1967, pp. 221-227.

[3] A. T. Drukarev and K. P. Yiu, "Performance of Error Correcting Codes on Channels with Memory", *IEEE Trans. Comm.,* vol COM-34, no.6, June 1986, pp. 513-521.

[4] R. McEliece and W. E. Stark, "Channels with Block Interference", *IEEE Trans. Inform. Theory,* vol IT-30, January 1984.

[5] S. Lin and D. J. Costello, *Error Control Coding*, Prentice Hall, Englewood Cliffs, New Jersey, 1983.

[6] J.L.Massey, "The How and Why of Channel Coding", *in Proc. Int. Zürich Sem. Digital Commun.*, Zürich, Switzerland, March 1984.

[7] J.L.Massey, *Applied Digital Information Theory*, Fach. Nr. 35-167, Lecture Notes, Federal Institute of Technology, Zürich, Switzerland, 1983

[8] J.L.Massey, *Coding for Everyday Channels*, Short Course Notes, ICCC 1973, Seattle, Washington.

[9] Andrew J. Viterbi and Jim K. Omura, *Principles of Digital Communication and Coding*, McGraw-Hill, New York, 1979.

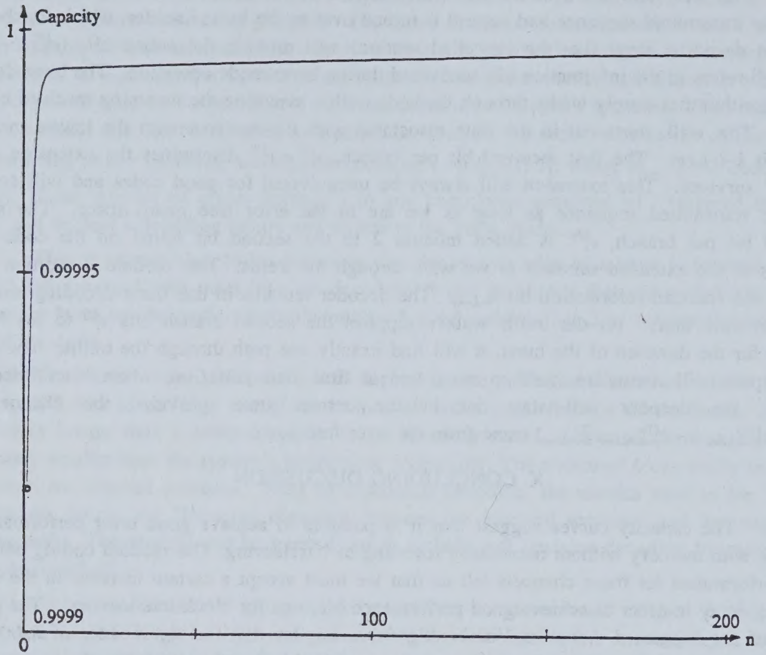


Fig. 4 Capacity of the magnetic recording channel reported in [3]
Bit error rate = 10^{-6}

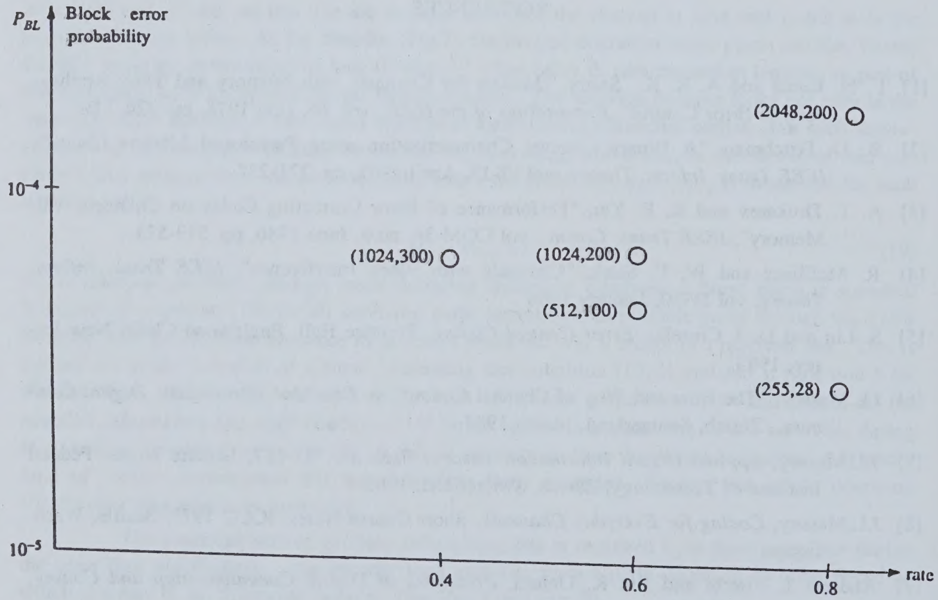
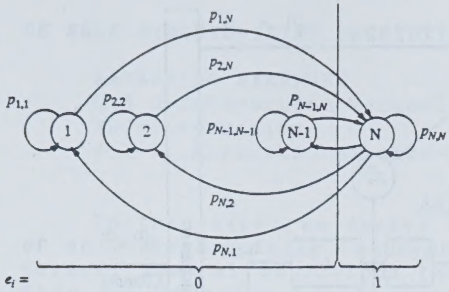


Fig.5. Error performance of (n,(n-k)/2) single burst error correcting codes on the magnetic recording channel reported in [3]; (n-k)/2 is the burst correcting limit of the code.


$p_{1,N}$
$p_{2,N}$
$p_{1,1}$
$p_{2,2}$
$p_{N-1,N}$
$p_{N-1,N-1}$
$p_{N,N}$
1
2
N-1
N
$p_{N,2}$
$p_{N,1}$
$e_i =$
0
1

$$P = \begin{bmatrix} p_{1,1} & 0 & \cdot & 0 & p_{1,N} \\ 0 & p_{2,2} & & & p_{2,N} \\ & 0 & & & \\ 0 & & & p_{N-1,N-1} & p_{N-1,N} \\ p_{N,1} & p_{N,2} & \cdot & p_{N,N-1} & p_{N,N} \end{bmatrix}$$

Fig.1. Simplified Fritchman model for a burst noise channel

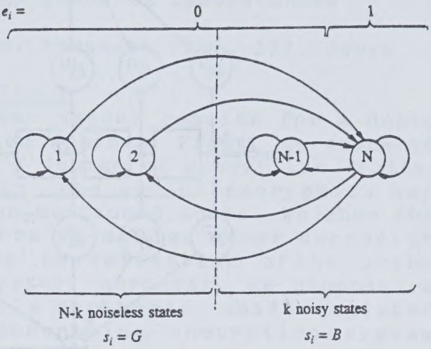
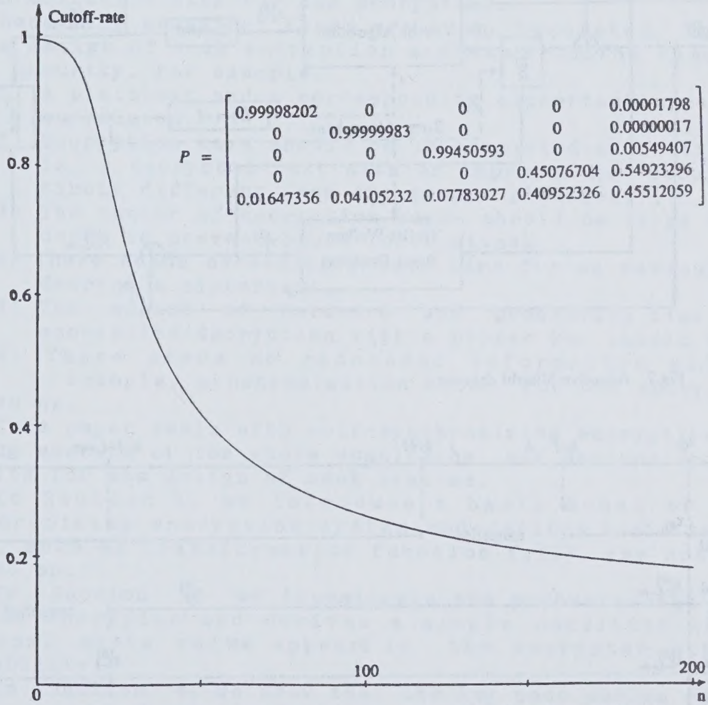

$e_i =$
0
1
1
2
N-1
N
N-k noiseless states
$s_i = G$
k noisy states
$s_i = B$

Fig.2. Side Information on the Fritchman channel model



$$P = \begin{bmatrix} 0.99998202 & 0 & 0 & 0 & 0.00001798 \\ 0 & 0.99999983 & 0 & 0 & 0.00000017 \\ 0 & 0 & 0.99450593 & 0 & 0.00549407 \\ 0 & 0 & 0 & 0.45076704 & 0.54923296 \\ 0.01647356 & 0.04105232 & 0.07783027 & 0.40952326 & 0.45512059 \end{bmatrix}$$

Fig.3. Computational cutoff rate of the magnetic recording channel reported in [3]
Bit error rate = 10^{-6}

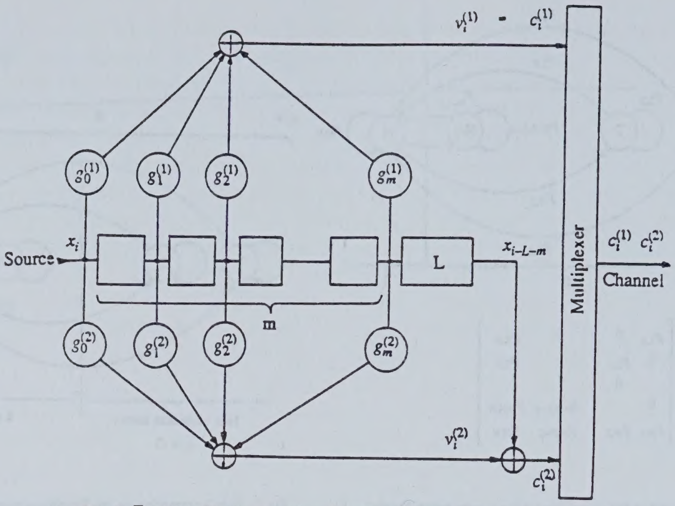

$v_i^{(1)} = c_i^{(1)}$
$g_0^{(1)}$
$g_1^{(1)}$
$g_2^{(1)}$
$g_m^{(1)}$
Source
x_i
L
x_{i-L-m}
m
Multiplexer
$c_i^{(1)}$ $c_i^{(2)}$
Channel
$g_0^{(2)}$
$g_1^{(2)}$
$g_2^{(2)}$
$g_m^{(2)}$
$v_i^{(2)}$
$c_i^{(2)}$

Fig.6. Encoder for the Adaptive decoding scheme for a (2,1,m) code.

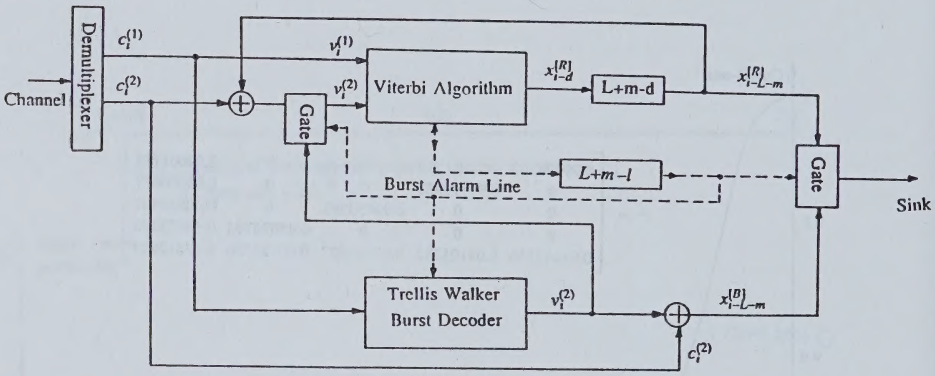

Demultiplexer
Channel
$c_i^{(1)}$
$c_i^{(2)}$
$v_i^{(1)}$
$v_i^{(2)}$
Gate
Viterbi Algorithm
$x_{i-d}^{[R]}$
L+m-d
$x_{i-L-m}^{[R]}$
Burst Alarm Line
L+m-l
Gate
Sink
Trellis Walker Burst Decoder
$v_i^{(2)}$
$x_{i-L-m}^{[B]}$
$c_i^{(2)}$

Fig.7. Adaptive Viterbi decoder

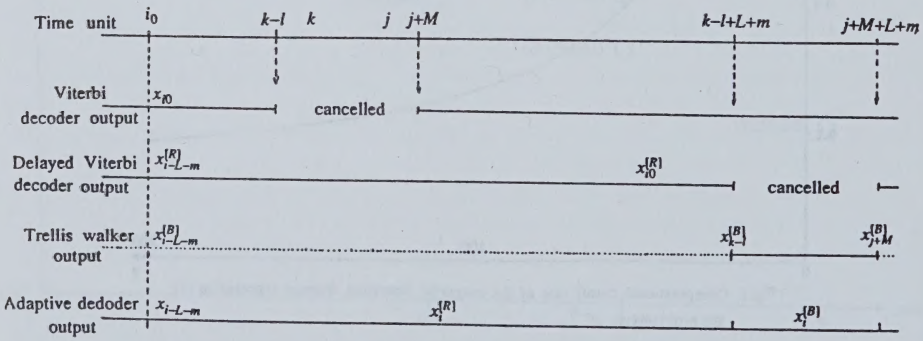

Time unit
i_0
$k-l$
k
j
$j+M$
$k-l+L+m$
$j+M+L+m$
Viterbi decoder output
x_{i0}
cancelled
Delayed Viterbi decoder output
$x_{i-L-m}^{[R]}$
$x_{i0}^{[R]}$
cancelled
Trellis walker output
$x_{i-L-m}^{[B]}$
$x_{k-l}^{[B]}$
$x_{j+M}^{[B]}$
Adaptive dedoder output
x_{i-L-m}
$x_i^{[R]}$
$x_i^{[B]}$

Fig. 8. Timing diagram for the Adaptive Viterbi decoder

ON SELF-SYNCHRONIZING ENCRYPTION SYSTEMS

KATSUHIRO NAKAMURA
C & C Information Technology Research Laboratories
NEC Corporation
4-1-1, Miyazaki, Miyamae-ku, Kawasaki City, 213, Japan

ABSTRACT

In this paper, we derive some useful results for a design of self-synchronizing encryption systems. First, we summarize several general conditions for a design of encryption systems. Then, we describe how to design the set of encryption keys taking account of the conditions mentioned above. We show that the key set design results in the so-called error correcting /detecting code design by the investigation of the probabilistic behavior of the encryptor. Moreover, we propose an encryption system with multiple decmposed shift-registers to get more robust self-synchronizing encryption systems satisfying the above conditions..

1. INTRODUCTION

Self-synchronizing encryption systems are popular in modern digital data communication systems since they needs no synchronization bits for the encryption.

Generally speaking, there are many requested conditions for a design of such encryption systems from the viewpoint of data security. For example,

(1) A plaintext and a corresponding ciphertext should be uncorrelated,
(2) Encryption keys should be uncorrelated each other (That is, a decrypted text with an improper key should be entirely different from the proper plaintext),
(3) The number of encryption keys should be large enough in order to prevent brute-force attack,
(4) There needs exceedingly much time for an eavesdropper to decrypt a ciphertext,
(5) The amount of hardware and processing time for the encryption/decryption with a proper key should be small,
(6) There needs no redundant information bits (for example, synchronization bits) for the encryption,

and so on.

This paper deals with self-synchronizing encryption systems taking account of the above conditions and derives some useful results for the design of such systems.

In Section 2, we introduce a basic model of a self-synchronizing encryption system and defines basic terminologies, such as transformation function $f_i(X)$, key code $K=\{f_i\}$ and so on.

In Section 3, we investigate the probabilistic behavior of the encryptor and derives a simple condition that each internal state value appears in the encryptor with equal probability.

In Section 4, we show that the key code design results in the so-called error correcting /detecting code design using the result of Section 3.

In Section 5, we propose an encryption system with multiple decomposed shift-registers to get more robust self-synchronizing encryption systems satisfying above conditions.

2. BASIC MODEL OF SELF-SYNCHRONIZING ENCRYPTION SYSTEM

Figure 1 shows a basic model of a self-synchronizing encryption system. In Fig.1, each symbol indicates the following.

$d(t)$: plaintext datum at time t

$c(t)$: ciphertext datum at time t

$X(t)$: internal state value X at time t, where

$$X = x_0 + x_1*2 + x_2*2^2 + ... + x_{m-1}*2^{m-1} \quad (1)$$

$f_i(X)$: transformation function of internal state value X to one bit value 1 or 0.

In the system shown in Fig.1, the i-th encryption key corresponds to the transformation function f_i.

Since the function f_i can be represented as a code pattern, $(f_i(0), f_i(1), ..., f_i(2^m-1))$, of length $M = 2^m$, the number of possible code patterns f_i's is 2^M. Among these 2^M code patterns, we must select appropriate code patterns as the encryption keys so that the conditions listed in Section 1 may be satisfied. We call such a set $K=\{f_i\}$ of selected code patterns a key code K of the encryption system. How to design the key code is shown in Section 4.

Now, the system shown in Fig.1 apparently needs no redundant bits for the encryption. Even if an error has occured in the channel(or memory), the synchronization recovers automatically after m bits time-intervals. That is, the internal state of the decryptor coincides the state of the encryptor. Therefore, the system satisfies the condition (6) in Section 1.

Before we proceed to consider the conditions (1)～(5)for the system in Fig. 1, we investigate the probabilistic behavior of the encryptor in the next section. The result of the next section is very useful for a design of the system satisfying the conditions in Section 1.

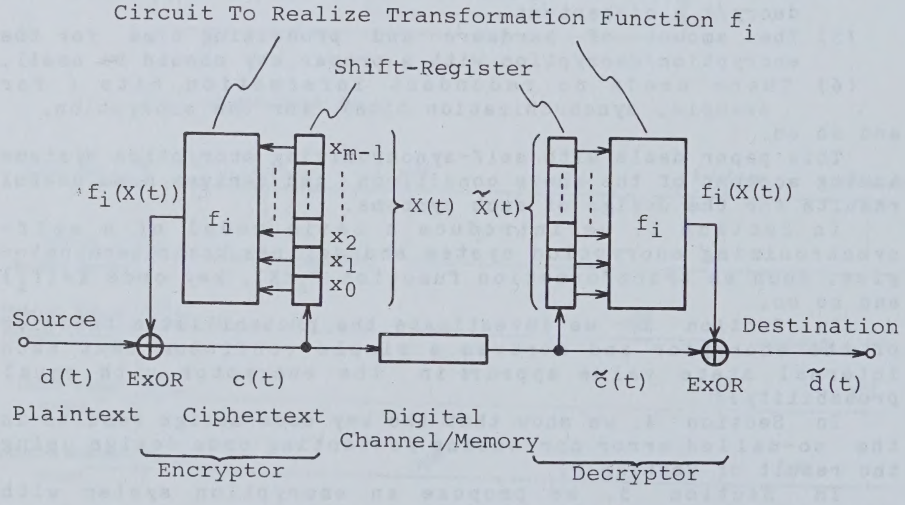

Circuit To Realize Transformation Function f_i
Shift-Register
$f_i(X(t))$
f_i
x_{m-1}
x_2
x_1
x_0
$X(t)$
$\tilde{X}(t)$
f_i
$f_i(X(t))$
Source
Destination
$d(t)$
ExOR
$c(t)$
$\tilde{c}(t)$
ExOR
$\tilde{d}(t)$
Plaintext
Ciphertext
Digital Channel/Memory
Encryptor
Decryptor

Fig.1 Basic Model of Self-Synchronizing Encryption System

3. Probabilistic behavior of the encryptor

Let the source in Fig.1 be a binary discrete memoryless source with occurrence probability p ($\neq 0$) for the symbol 1 and $q=1-p$ for the symbol 0. Then, the transition process of internal state $\{X(t)\}$ of the encryptor can be regarded as a finite simple Markov process satisfying the following equation.

$$X(t+1) = 2*X(t) + (\, f_i(X(t)) \oplus d(t)\,) \quad (\bmod M=2^m), \tag{2}$$

where $\oplus$ denotes mod 2 operation.

For example, if $f_i(X(t))=1$, $X(t+1)=2*X(t) \pmod M$ with probability p and $X(t+1)=2*X(t)+1 \pmod M$ with probability q. Then, the following theorem holds good.

Theorem 1 [2] The stationary occurrence probability of each state X all becomes $1/2^m$ when either one (or both) of the following conditions holds good.

$$(1)\ p=q=1/2 \tag{3}$$

$$(2)\ f_i(X) \oplus f_i(X+2^{m-1})=1 \quad (\bmod 2), \tag{4}$$

$$\text{where } 0 \leq X < 2^{m-1}$$

Proof : In the Equation (2), each probability that ($f_i(X) \oplus d(t)$) equals 0 or 1 does not equal to 0 since the probability p is not 0. Therefore, in the transition diagram for the probability process $\{X(t)\}$, there exists a path from each state X to any state X'. Moreover, there exists a state X having a self-loop in the transition diagram. (For example, the state $X = 2^m-1$ has a self-loop.) Hence, the probabilistic process $\{X(t)\}$ is irreducible and non-periodic.

Therefore, if we can show that the sum of probabilities of input edges to each state X is 1, the proof is completed[3][4].

There are only two input edges to each state X in the state transition diagram. One edge stems from the state $Y=X/2$ and the other from the state $Y' = X/2 + 2^{m-1}$, where $X/2$ shows the integer part of X divided by 2. On the other hand, the probability of each input edge is p or q from Eq.(2). Therefore, the condition that the sum of probabilities of input edges to each state is 1 is given as Eq.(3) or (4). Q.E.D.

4 Key code design for the encryption system

In this section, we derive a design method for a key code $\{f_i\}$ of the self-synchronizing encryption system taking account of the conditions (1) and (2) in Section 1.

First, we define reversed-bits rate of a binary sequence A to another binary sequence B of the same length as the ratio of the number of different bits between the sequence A and B to the length of the sequence A (or B).

Taking account of the conditions (1) and (2) mentioned above, we adopt the following design criteria [2].

(1)' The reversed-bits rate ε_1 of a plaintext to the corresponding ciphertext should be around 1/2.

(2)' The reversed-bits rate ε_2 of a plaintext to a decrypted text with an improper key should be around 1/2.

If we use only f_i's satisfying Eq.(2), it becomes very simple to design a key code $\{f_i\}$ satisfying the criteria (1)' and (2)'.

Figure 2 shows a code pattern f_i satisfying Eq.(2). In Fig.2, $\overline{f_i}'$ shows a reversed code pattern of f_i' of length 2^{m-1}. That is, $\overline{f_i}'(X) = f_i'(X) \oplus 1 \pmod 2$.

Figure 3 shows a circuit to realize a transformation function $f_i(X)$ satisfying Eq.(2).

Now, if we use a function f_i satisfying Eq.(2), all state values (X's) appear with equal probability $1/2^m$ in the Shift Register of the encryptor. Therefore, the reversed-bits rate ε_1 is equal to $\omega_i/2^m$, where ω_i is the number of 1's in the code pattern f_i. From Fig. 2, $\omega_i = 2^{m-1}$. Thus the following theorem holds good.

Teorem 2 Let each code pattern f_i of a key code $K=\{f_i\}$ satisfy Eq.(2). Then, reversed-bits rate ε_1 of a plaintext to the corresponding ciphertext becomes 1/2.

Next, we consider the criterion (2)' described above. Let a key code pattern used at the encryptor be f_i and that used at the decryptor be f_j. Then, if there is no channel (or memory) errors,

$$c(t) = d(t) \oplus f_i(X(t)), \quad \overset{*}{d}(t) = c(t) \oplus f_j(X(t)) \pmod 2, \tag{5}$$

where $\overset{*}{d}(t)$ denotes a decrypted text bit with f_j.

$$\text{Therefore,} \quad d(t) \oplus \overset{*}{d}(t) = f_i(X(t)) \oplus f_j(X(t)) \pmod 2. \tag{6}$$

Equation (6) shows that if $f_i(X(t)) = f_j(X(t))$ at time t, $d(t) = \overset{*}{d}(t)$ and $d(t) \neq \overset{*}{d}(t)$ if $f_i(X(t)) \neq f_j(X(t))$ at time t.

Moreover, each state value ($X(t)$) has the same occurrence probability $1/2^m$ since f_i satisfies Eq.(2).

Therefore, we can derive the following theorem from the above arguments.

$$f_i : (\underbrace{f_i(0), f_i(1), \quad , f_i(2^{m-1}-1)}_{f_i'}, \underbrace{f_i(2^{m-1}), \quad f_i(2^m-1)}_{\overline{f_i}'})$$

Fig.2 Code Pattern f_i Satisfying Eq.(4)

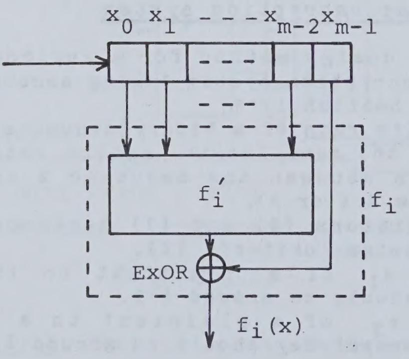


Fig.3 Circuit To Realize Transformation Function f_i Satisfying Eq.(4)

Theorem 3 Let the Hamming distance $d(i,j)$ between each code pattern f_i and another pattern f_j satisfy the following equation.

$$(1/2-\varepsilon)*M < d(i,j) < (1/2+\varepsilon)*M, \quad \varepsilon \approx 0, \tag{7}$$

where M is the code length.

Then, reversed-bits rate ε_2 of a proper plaintext to the decrypted text with an improper key code pattern f_j (proper key code pattern is f_i) satisfies the following equation.

$$|\varepsilon_2 - 1/2| < \varepsilon$$

From Theorem 3, we can see that the key code design for the self-synchronizing encryption system has resulted in the design of the so-called error correcting/detecting code design.

Since $f_i' \oplus f_j' = \overline{f_i'} \oplus \overline{f_j'}$, we have only to design a code $K'=\{f_i'\}$ of length $M'=2^{m-1}$ with the property that the distance $d(i,j)$ between any two codewords (code patterns) f_i' and f_j' satisfies Eq.(7).

Example : Suppose a (4096,82 ~~4014~~,1776) extended BCH code as a key code $K'=\{f_i'\}$. The number of key code patterns (f_i''s) is $2^{4096-4014} \approx 4.84*10^{24}$. Since the minimum and maximum distance of this code are 1776 and 2320 respectively, the minimum and maximum of reversed-bits rates ε_2 are 0.434 and 0.566 respectively. Therefore, for the obtained key code, $|1/2-\varepsilon_2| < 0.066$.

5. Encryption System with Multiple Decomposed Shift-Registers

If we choose a key code K satisfying Eq.(2) and Eq.(7), we can get a self-synchronizing encryption system satisfying to fairly extent the conditions (1) and(2)listed in Section 1. In order to get the system satisfying the conditions (3) and (4), we must make the number m of stages of the Shift-Registers in Fig.1 large. However, if the number m is too large, the condition (5) becomes hard to be satisfied. In order to resolve this problem, we propose a self-synchronizing encryption system with multiple decomposed Shift-Registers as shown in Fig.4 and Fig.5. The system shown in Fig. 5 is a generalized version of the system in Fig.4.

In Fig.4, shift-registers are represented by two internal state values , $X(t)$ and $Y(t)$, which satisfy the following equation.

$$Y(t+1) = 2*Y(t) + (f(X(t)) \oplus d(t)) \pmod{2^{m'}}, \tag{8}$$

$$X(t+1) = 2*X(t) + (f(X(t)) \oplus d(t) \oplus g(Y(t))) \pmod{2^m}, \tag{9}$$

where $\oplus$ denotes mod 2 operatipon.

From Eq.(9), Theorem 1 holds good for the state X in Fig.4 by the similar arguments given in the proof of Theorem 1.

(Moreover, from Eq.(8), the stationary occurrence probability of each state Y all becomes $1/2^{m'}$ on the condition that $\mathrm{Prob}\{f(X(t))=1\} = \mathrm{Prob}\{f(X(t))=0\} = 1/2$.)

Therefore, we can design an encryption system shown in Fig.4 which satisfies the design criteria (1)' and (2)' described in Section 4 by choosing a key code $K=\{f_i\}$ satisfying Eq.(2) and Eq.(7). In this case, the part encircled with a dotted line in Fig.4 is designed to become a random bit sequence generator with high non-liniearity(and non-predictability)[5].

Similar arguments can be made for the encryption system in Fig.5. In Fig.5, the part encircled with a dotted line can be constructed as a more powerful random bit sequence generator than that in Fig.4, since more shift-registers and more transformation functions (g_i's) can be used.

By taking account of the above discussions, we can construct a self-synchronizing encryption system satisfying the conditions (1)∼(6) in Section 1 to a great extent.

6. Conclusions

We have derived some useful results for a design of a self-synchronizing encryption system.

The key code design has resulted in the design of the so-called error correcting/detecting code design by the investigation of the probabilistic behavior of the encryptor.

We have also discussed a design of a robust self-synchronizing encryption system and proposed a system with multiple decomposed shift-registers which to a great extent satisfies several requested general conditions for an encryption system.

However,the analysis has been made based on the stationary occurrence probabilities of the internal states. Thus, it may be preferable to use a linear data scrambler before the encryptor and a linear data descrambler after the decryptor in order to realize the stationary probabilities more rapidly.

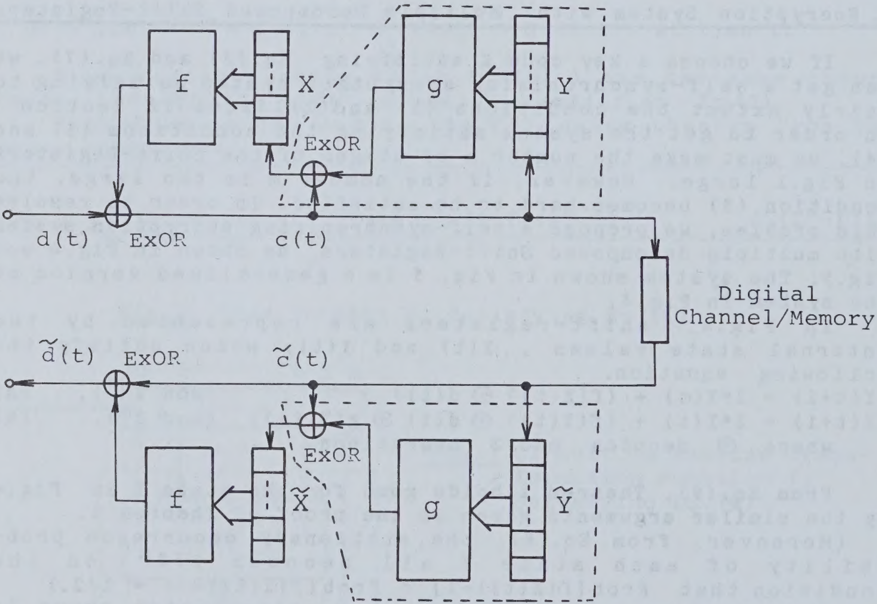


Fig.4 Self-Synchronizing Encryption System with Decomposed Shift-Registers

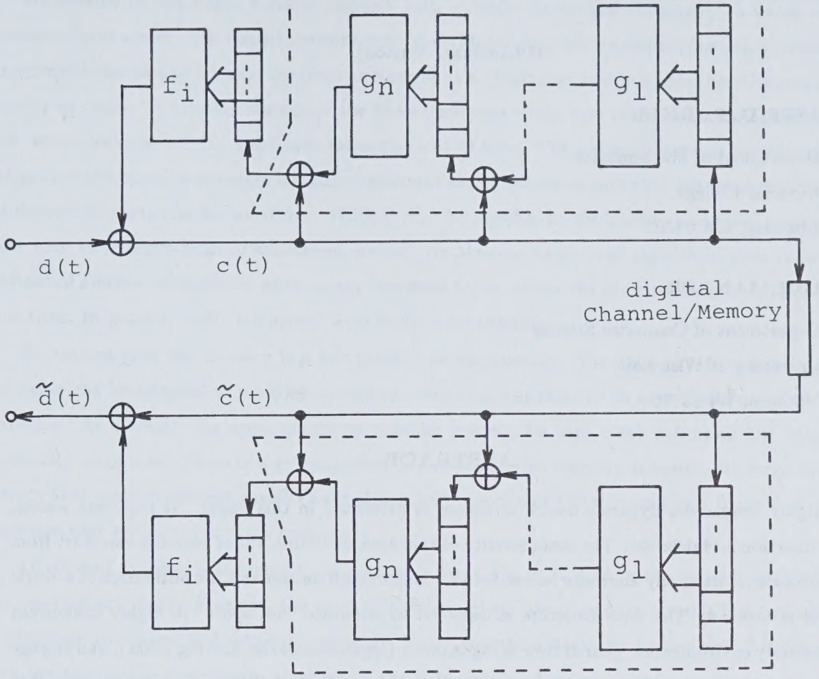


Fig.5 Self-Synchronizing Encryption System with Multiple Decomposed Shift-Registers

7. Ackowledgement

The author would like to thank Mr. Matsuo, Dr.Akaiwa Mr.Sabato and Dr.Okamoto for their discussions. He also would like to thank Mr.Kato, Mr.Goto and Mr.Managaki for their encouragement.

8. References

[1] S.W.Golomb, "Shift Register Sequences", Aegean Park Press, Revised edition, 1982
[2] K.Nakamura, "Some considerations on a simple self-synchronizing encryption system", Proc. of the 3rd Symposium on Information Theory and Its Applicationπ, November, 1980 (in Japanese)
[3] T.Kitagawa, ed., "Markov Process", Kyoritsu Shuppan, 1967, (in Japanese)
[4] J.G.Kemeny and J.L.Snell, "Finite Markov Chains", D.Van Nostrand Company, Inc., 1960
[5] J.B.Kam and G.I.Davida "Structured Design of Substitution-Permutation EncryptionNetworks", IEEETrans. on Computers, Vol.C-28, No.10, October 1979

THE SIBLING TRIE: A HIGHLY CONCURRENT SEARCH STRUCTURE ‡

(Preliminary Version)

JEFF D. PARKER

Department of Mathematics
Amherst College
Amherst, MA 01002

UDI MANBER

Department of Computer Science
University of Wisconsin
Madison, WI 53706

ABSTRACT

A highly concurrent dynamic search structure is presented in this paper. It supports search, update, insertion and deletion. The main novelty of this data structure is that searches can start from any node. As a result, many alternate routes are provided to each datum and the bottleneck of a single root node is avoided. The data structure is designed to minimize "hot spots" in highly concurrent shared memory environments. Search time is logarithmic (regardless of the starting point), and storage is proportional to the number of data items and it is independent of the number of processes that access the data structure.

1. INTRODUCTION

We consider multicomputers consisting of many asynchronous independent processors with some type of shared memory. The study of such multicomputers has been a very active area of research in the last few years. Several such multicomputers are now being marketed, ranging from two to more than a hundred processors. It is hard to predict future trends, and, in particular, it is not clear what is the practical limit to the number of processors in such systems. However, it seems likely that machines with thousands of processors may be offered in the future, at least for some applications. One of the major problems of using such machines will be the design of highly concurrent data structures to support highly parallel execution. Since one cannot predict the exact architecture of future multicomputers, it is more beneficial to study general "generic" models. Of particular interest are studies in inherent limits of parallelism.

---

‡ This research was supported in part by the National Science Foundation under grants MCS-8303134 and DCR-8451397.

We assume in this paper a multicomputer with a single shared global memory to which many processors have access. The shared memory may be divided among the processors (i.e., each processor is responsible for part of it), or it may be one integral unit. Each processor may also have its own local memory or cache. A process can access the shared memory using four atomic actions, called steps: read, write, lock, or unlock. Each step takes one unit of time. The order of access is unpredictable. The goal of this paper is to design a highly concurrent search structure such that the concurrency does not depend on particular architectures. While it may be possible to implement concurrent reads (and there were even suggestions of concurrent writes), we attempt to prevent algorithms from requiring substantial number of instances where many processes try to access the same memory location at the same time. In general, such "hot spots" tend to limit concurrency.

We assume that the memory is a fast random access memory. The data structure presented in this paper can be adapted to work for secondary memories, but most of its advantages occur for fast memories. As a result, the applications we consider are not for very large databases but only for moderately large ones. There is a growing effort to develop shared memory systems with large shared memory that support efficient parallel algorithms. Such algorithms will require highly concurrent data structures that are stored in main memory.

All efficient dynamic search structures are based in one way or another on the use of trees (hashing schemes that are highly dynamic, like extendible hashing, are also based on trees). The tree allows the structure to expand and shrink as necessary. Every operation starts at the root and it is directed down the tree to the appropriate place where the data resides. Having a single shared root is not a major problem if the number of processors accessing the tree at the same time is small. Even if many processes access the tree at the same time, it may still be possible to replicate the top of the tree and store it in local memory (or cache). The chances of colliding further down the tree are reduced, for unless the processes are looking for similar keys, they will be scattered throughout the tree. This solution may require substantial local memory from all processors. Furthermore, if the data structure is subject to frequent insertions and deletions, then the top of the tree may be altered, requiring a protocol to ensure the consistency of all copies.

We present in this paper a new dynamic data structure, called *sibling tries,* which allows multiple processes to search, insert, update, and delete shared data without introducing hot spots. Sibling tries, while based on trees, are a special kind of *graph*. Every node is reachable from every other node, and searches can start anywhere. Many alternate routes are provided to each datum, and the bottleneck of a single root node is avoided. Each data item in a sibling tree corresponds to constant amount of storage, independent of the number of processors sharing the data structure. For an efficient search, the graph should have the following properties:

1. The *diameter* of the graph (i.e., the maximal shortest distance between any two nodes) should be

small, preferably logarithmic in the number of nodes.

2. The graph should be strongly connected so that each node is reachable from any other node.
3. The set of paths from any one node to all other nodes should correspond to an efficient search structure.
4. There should be no bottlenecks – no node should be included in too many search paths.
5. The graph should be easily expandable, i.e., it should support efficient insertions and deletions.

We present in this paper a scheme that satisfies all the requirements listed above. The main idea of our scheme is to use graphs similar to ones used in interconnections networks. Interconnection networks have been studied extensively in the last few years (see for example [12]). They are used to connect computers to minimize the time for communication, the cost of the connection, and other factors. Interconnection graphs generaly satisfy requirements 1, 2, and 4. Our solution to the problem of expanding the graph and incorporating a search structure is to tie together a series of graphs of increasing size with a trie structure. This is explained further in section 3.

2. RELATION TO OTHER WORK

This paper brings together two areas in the design of parallel systems: concurrent data structures and interconnection networks. Both areas have been studied extensively and we cannot list here all relevant work. Most of the work on concurrent data structures concentrates on concurrent B-trees and their variations [7], as B-trees are the most common data structure used in databases. Our emphasis is on databases residing in main memory. Ellis presented algorithms for concurrent search and insert in AVL trees [1], and 2-3 trees [2]. Efficient concurrent algorithms for regular and external binary trees without balancing were given in [10] and [9]. We use the algorithms in [9] as the basis for our design. Very highly concurrent algorithms for a simple data structure that supports restricted insertions and deletions without search were presented in [8].

Several variations of hashing were also studied in this context. Ellis has studied a concurrent version of extendible hashing [3] and linear hashing [4]. Both structures provide very quick access to large data bases, but both require that the search starts from one fixed point. Introducing independent processes to perform maintenance tasks in parallel played a major part in [1], [10], and [9].

3. DATA STRUCTURE

The first component of our data structure is a trie – a binary tree that implements a radix search [11]. The keys are bitstrings, and the parent of a node with key w holds a key which a proper prefix of w. Data is stored in both internal and external nodes. Each node of the trie is assigned a unique key. We define the *level* of a node as the length of the key. We will informally use the key, the node containing the key, and a pointer to the node interchangeably. There may be internal nodes that contain no data, but distinguish different keys with a common prefix. The trie is only our starting point – like the tree, it has a bottle-neck at the root. We link together nodes with keys of the same length into a *sibling graph*. We use the DeBruijn graphs [5], though other graphs can also be used. These graphs are used as hints: the trie always defines the current state of the data structure. The sibling graphs have a fixed form: the graph at level k is a subgraph of the DeBruijn graph on 2^k vertices. These subgraphs need not be complete or even connected, as each node is a part of the trie, a connected graph. In practice, the graphs close to the root will be the entire DeBruijn graph, which are strongly connected, while those deeper in the trie will be incomplete and disconnected.

Each node has the following variables:

1 The **key**, a string of bits,

2 5 pointers: **parent**, **child[0]**, and **child[1]** (which we sometimes call **left** and **right**), and two pointers to siblings **sib[0]** and **sib[1]**,

3 A pointer **data** to the information associated with the node,

4 a **deleted** flag, and

5 a lock field.

The DeBruijn graph of degree k on s vertices is defined as follows. We number the vertices from 0 to $s-1$, and insert an edge from vertex number v to vertex number w if and only if $w = (v \times 2 + i) \ mod\ s$, where $0 \leq i < k$. Thus in the DeBruijn graph of degree 2 on 64 vertices, the vertex 001010 has two edges directed outward: the edges (001010, 010100) and (001010, 010101). DeBruijn graphs are quite easy to construct and traverse. The structure may be calculated by shifting if the size s is a power of two, so we do not waste space storing the shape of the graph. The sibling graphs need small diameter, to allow rapid traversal. We could decrease the diameter by increasing the degree. However, this would increase the storage requirements of each node and the amount of work needed to update the graphs.

As we mentioned earlier, the sibling graphs connecting a given level in the trie need not be complete. They will be complete only if all the keys in that level are present. A key is present if it is either part of the data, or it is needed to distinguish keys below it. Let $full(k,n)$ be the probability that a trie T with n keys of length greater than k has a complete level at depth k. If $n \geq 2^{k+1}$, a standard application of inclusion-exclusion shows that

$$full(k,n) = \sum_{i=0}^{2^{k+1}-1} (-1)^i \left(\frac{2^{k+1}-i}{2^{k+1}}\right)^n \binom{2^{k+1}}{i}.$$

For example, 95% of all random tries with one thousand or more elements will have a complete level 6; 99% of all random tries with 27,000 or more elements will have a complete level 10.

4. ALGORITHMS

The concurrent algorithms for manipulating sibling tries are simpler than binary trees algorithms since the path from the root to any key is fixed. Only the length of such a path is variable and depends on the number of prefixes of the key that are currently in the trie. It is very easy to verify that a certain node is the correct node, in contrast to regular search trees where keys do not give any information on the structure of the tree. In addition, the existence of the sibling graphs simplifies the concurrency control. When several processes insert and delete concurrently, a search process may end up on a path that has been changed. If only one search path exists to any node, one has to construct recovery paths to ensure that search processes are not lost. In our scheme, since one can get to any node from any other node, recovery paths are already "built in". We only have to ensure that the trie remains strongly connected.

Many operations required for the algorithms need not be carried out immediately. We leave these operations to *maintenance* processes. This includes maintaining the sibling graphs, removing a node that was logically deleted by the user processes, and garbage collection. The maintenance processes may use the same processors, or dedicated processors. We omit many of the details of the maintenance processes algorithms in this preliminary version; the techniques are similar to the ones given in [10, 9]. We briefly describe the main algorithms.

4.1 *Search*

The search algorithm takes two input parameters: the search key, and a pointer p to a starting position in the trie. As was mentioned earlier, any node can serve as a starting position. We use the end node of the previous operation by this process (or processor) as the starting point for the next operation. The goal is to find the node with the search key, if it exists, or otherwise, to find a node where an insertion of the search key can take place. In either case we call the node the *goal node*. The search process returns the goal node locked. The search algorithm consists of either traversing the sibling graphs or traversing the trie (up or down). First, we traverse the sibling graph until either 1) we find the goal node, 2) we find an ancestor of the goal node, 3) we find a descendant of the goal node, or 4) the sibling graph is incomplete and the traversal is blocked as a result of it. In case 1 we are done, in case 2 and 3 we simply move up or down in the trie, and in case 4 we go up the trie and start again. Nodes are not required to be locked during the search. The only locking is done at the end when the correct node is found (we also check that it is correct after it is locked). This is similar to the way locking is done in [6, 10].

4.2 *Insertion*

To insert x, we first search to find the goal node p, which is returned locked. Tries may have empty nodes, that is, nodes that are used to distinguish items which share a common prefix. The data pointer of an empty node is nil. Thus there are two possibilities after a successful search. The node p

may be empty, or it may already be in use. If the data pointer is nil, we alter it to point to our new information, and release the locks. If the data pointer is not nil, the key is already in use, and we release locks.

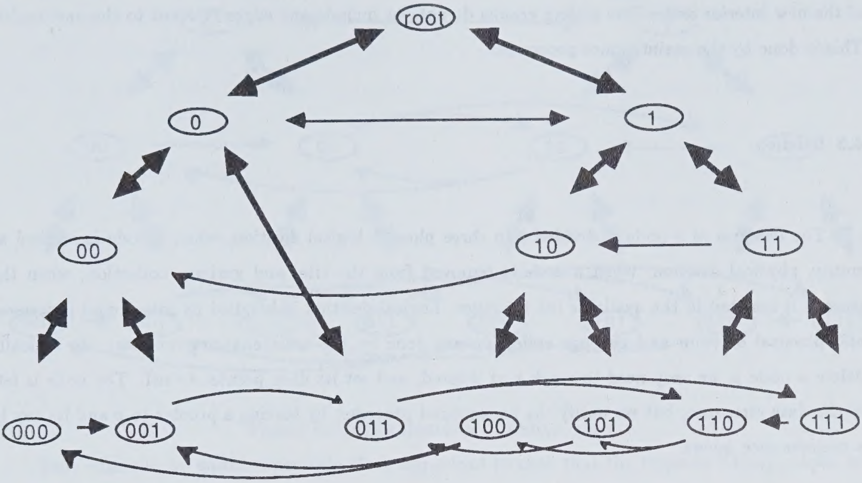

root
0
1
00
10
11
000
001
011
100
101
110
111

Figure 4.1: Before insertion of 010

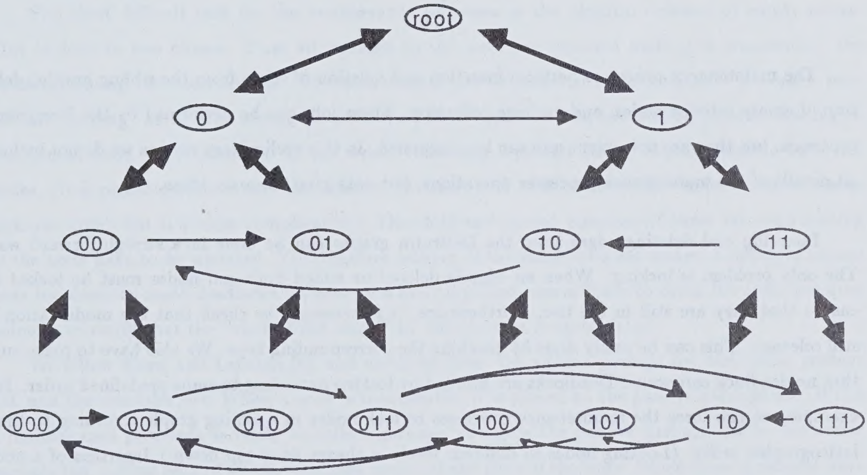

root
0
1
00
01
10
11
000
001
010
011
100
101
110
111

Figure 4.2: After insertion of 010

If the search is not successful, p is the lowest potential parent for x currently in the trie. Consider the task of inserting 010 in figure 4.1. When the search returns, we will hold a lock on $p = 0$. Note that *p.right* is in use: we say that *p.right* points to an *impostor*. To insert, we lock the impostor, retaining the lock on *p*. We insert a new interior node, 01, between the impostor and its parent, and update the corresponding child and parent pointers. The impostor and the new node x will be on either side of the new interior node. The sibling graphs do not yet include any edges incident to the new nodes. This is done by the maintenance processes.

4.3 *Deletion*

The deletion of a node is divided into three phases: logical deletion, when a node is marked as empty, physical deletion, when a node is removed from the trie, and garbage collection, when the storage is inserted in the available list for reuse. Logical deletion is handled by foreground processes, and physical deletion and garbage collection are done by the maintenance processes. To logically delete a node p, we only need to mark p at deleted, and set its data pointer to **nil.** The node is left in the data structure, but we notify the background processes by leaving a pointer to p and its key in a maintenance queue.

4.4 *Maintenance Processes*

The maintenance processes perform insertion and deletion of edges from the sibling graphs, deletion of empty interior nodes, and garbage collection. These jobs can be performed by the foreground processes, but they are not urgent and can be postponed. In this preliminary version we do not include all details of the maintenance processes operations, but only give the main ideas.

Inserting and deleting edges from the DeBruijn graphs can be done in a straightforward way. The only problem is locking. When an edge is deleted or added both end nodes must be locked to ensure that they are still in the trie. Furthermore, it is necessary to check that the modification is still relevant. This can be easily done by checking the corresponding keys. We also have to make sure that no deadlock can occur. Deadlocks are avoided by locking according to some predefined order. For example, we can force the maintenance processes to lock nodes in a sibling graph according to their lexicographic order. (Locking nodes in different levels is always done top down.) Insertion of a node **can lead to 4 new edges (a new key may require two new nodes)**. Figure 4.3 shows the trie of figure 4.2 after maintenance processes have inserted the edges needed.

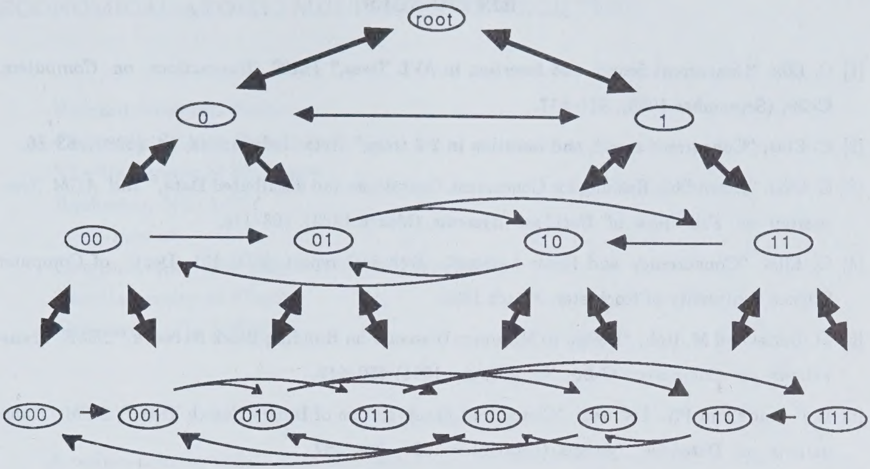


Figure 4.3: After insertion of sibling edges

Each edge can be added separately. It is important to note that the edges in sibling graphs are not required for the correctness of the algorithms – they only improve concurrency. Thus adding or deleting such edges can be postponed.

The most difficult task for the maintenance processes is the physical deletion of empty nodes. This is done in two phases. First all pointers to the node are removed making it inaccessible; the pointers leaving the node remain. The node cannot yet be removed since some slow processes may still be "looking" at it. The second phase is the actual removal. The two sibling pointers leading to the deleted node can be removed in any order. Removing a pointer (edge) requires locking both end nodes. (It is possible to have only the deletion processes lock both nodes while the insertion processes lock one node, but it's more complicated.) The child and parent pointers (if there are any) leading to the node have to be updated. This requires locking of the other involved nodes. Locking is always done top down to avoid deadlocks. Whenever a node is locked, one still has to check the corresponding pointers to verify that the "view of the world" by this process is up-to-date.

We follow Kung and Lehman [6], and use three lists: the *passive garbage list*, the *active garbage list*, and the *available list.* When a node is inaccessible, it is placed on the passive garbage list. When a maintenance processes is ready to collect garbage, it copies the passive garbage list to the active garbage list, taking note of all the processes active at the time of the copy. When these processes have terminated, the active garbage list is appended to the available list. These nodes are invisible to any current foreground process. There may be a reference to the deleted node in the maintenance queues, but each maintenance processes checks the keys before using any pointer, so it is safe to reuse the storage.

REFERENCES

[1] C. Ellis "Concurrent Search and Insertion in AVL Trees," *IEEE Transactions on Computers,* **C-29,** (September 1980), 811–817.

[2] C. Ellis, "Concurrent search and insertion in 2-3 trees," *Acta Informatica,* **14**, (1980), 63–86.

[3] C. Ellis, "Extendible Hashing for Concurrent Operations and distributed Data," *2nd ACM Symposium on Principles of Database Systems,* (March 1983), 106–116.

[4] C. Ellis, "Concurrency and linear hashing", Technical report #TR-151, Dept. of Computer Science, University of Rochester, March 1985.

[5] M. Imase and M. Itoh, "Design to Minimize Diameter on Building-Block Network," *IEEE Transactions on Computers,* **C-30,** No. 6, (June 1981) 439–442.

[6] H.T. Kung and P.L. Lehman. "Concurrent Manipulation of Binary Search Trees," *ACM Transactions on Database Systems,* (September 1980), 354–382.

[7] P. L. Lehman and S. B. Yao, "Efficient Locking for Concurrent Operations on B-Trees", *ACM Transactions On Database Systems* **6,** (December 1981), 650–670,

[8] U. Manber, "On Maintaining Dynamic Information in a Concurrent Environment," *Sixteenth Annual ACM Symposium on Theory of Computing,* (April 1984), 273–278. (To appear in *SIAM J. on Computing,* Nov. 1986.)

[9] U. Manber, "Concurrent Maintenance of Binary Search Trees", *IEEE Transactions on Software Engineering,* **SE-10,** No. 6, (November 1984), 777–784.

[10] U. Manber and R.E. Ladner, "Concurrency Control in a Dynamic Search Structure", *ACM Transactions on Database Systems,* **9,** No. 3, (September 1984), 439–455.

[11] R. Sedgewick, *Algorithms,* Addison-Wesley, Reading, MA, 1983.

[12] H. J. Siegel, *Interconnection networks for large scale parallel processing: theory and case studies,* Lexington Books, Lexington, MA, 1985.

ECONOMICAL ATOMIC MULTI-READER REGISTERS

Richard Newman-Wolfe
Computer Science Department
The University of Rochester
Rochester, NY 14627
now at
Computer and Information Sciences Department
The University of Florida
Gainesville, FL 32611

ABSTRACT

A writer-priority solution to the atomic *(r,1)*-CRWW problem is presented. It uses *(r+2)(2+B)* safe, multi-reader bits and *(r+2)r* safe, single-reader bits to implement a *b*-bit buffer shared among *r* readers and *1* writer. The writer never waits and the readers do not suffer lockout. This solves an open question of Lamport. This result is one extreme of a spectrum of solutions trading waiting for space. Using a multi-writer version of Lamport's wait-free regular register, a multi-reader, multi-writer, writer-priority, atomic register is built from regular bits.

INTRODUCTION

The concurrent reading while writing *(CRWW)* problem is sharing a data structure between several asynchronous processes ([Courtois, Heymans and Parnas'71],[Lamport'77],[Peterson'81,'83a and '83b],[Lamport'85]). It has the flavor of the mutual exclusion problem [Dijkstra'65], but has significant differences. In fact, early solutions to the problem simply used mutual exclusion, but this is overly restrictive.

The shared resource can be modeled as a buffer accessed by several processes of two types: readers and writers. Only writers may change the value of the buffer, which can be read simultaneously by many readers. Access to the buffer is controlled by some structure of concurrently accessed variables. In general, there are *w* writers and *r* readers, all of whom may be accessing the same bit at the same time. This is referred to as the *(r,w)-CRWW* problem.

Leslie Lamport has identified several categories of shared variable in terms of the interaction between one writer and one or more readers [Lamport'85]. These *safeness properties* are, in order of increasing strength, *safe, regular* and *atomic.* The power of a shared variable includes not only its safeness properties, but the number of each type of process that may access it. This paper extends the results by Lamport and Peterson in simulating powerful variables using weaker ones.

A *CRWW protocol* is a protocol access to a shared buffer for two kinds of processes, readers and writers, and that guarantees certain safeness properties for the buffer it implements. A CRWW protocol is *writer-priority* if the writer never has to wait on a reader [Lamport'77]. If neither type of process waits, the solution is called *wait-free*.

Lamport introduced the first writer-priority, atomic *(r,1)*-CRWW solution that used regular shared variables [Lamport'77]. The solution used only one buffer but had control variables that had to hold arbitrarily large values. Furthermore, the readers could starve. A wait-free solution was given by [Peterson'81], but this uses $r+2$ buffers and had some atomic control bits. Other, earlier solutions used very powerful synchronization primitives, or equivalently, powerful control variables in their implementation [Courtois, Heymans and Parnas'71].

The solutions presented here require only safe bits to implement atomic variables. The final solutions are *writer-priority* since the writer never waits on a reader [Lamport'77]. Unlike the solution of [Lamport'77], readers cannot be locked out. The writer never has to write more than one copy of the shared buffer and a few control bits per reader, and no reader has to write more than one control bit per read. This contrasts with the solutions given in [Peterson'81] where the writer may have to write up to $r+2$ copies of the shared buffer per write. The solutions given here yield tradeoffs between space (and potentially large amounts of work as in [Peterson'81]), and potentially having to wait for other processes.

A SPECTRUM OF (r,1)-CRWW SOLUTIONS

Peterson presents an algorithm in which neither readers nor writer waits using $r+2$ copies of the shared buffer [Peterson'79]. Peterson shows this number of buffers to be required for any wait-free solution. His construction produces an atomic buffer for one writer and r readers using $2r$ atomic single-reader bits; two atomic, r-reader bits; and $b(r+2)$ safe r-reader bits, where b is the buffer size. The writer must always make two copies of the buffer, and may have to make as many as $r+2$ copies. A difficulty of this construction is that it is not known how to make wait-free, atomic, r-reader bits from weaker variables.

Algorithm 1 requires $M(2+r+b)-1$ safe bits to implement a b-bit atomic buffer shared among r readers and one writer. More control bits are used than in Peterson's solution, but the control bits are weaker. The algorithm provides a spectrum of solutions that trade space for waiting, and implements a writer-priority solution if $r+2$ buffers are used.

The solution uses an $(M-1)$-bit regular register written only by the writer, that selects the current buffer (see Fig. 1). The selector register is implemented by Lamport's wait-free, multi-reader, regular register from safe bits [Lamport'85]. For each buffer we have control bit written by the writer and r control bits written by the readers (one per reader per buffer). If each buffer has b bits, the total number of safe bits used for the algorithm is $M(2+r+b)-1$.

Theorem 1: *Algorithm 1 implements a lockout- and deadlock-free, atomic, r-reader, single-writer buffer using safe bits.*

Correctness is shown by the inclusion of a basic, unfair mutual exclusion algorithm. The writer always defers to a reader, so readers are not locked out and deadlock is impossible. The writer can wait on each reader at most once per write, so it is not locked out either.

Lemma 2: *Algorithm 1 for r readers will make the writer wait on at most* $\lfloor r/(M-1) \rfloor$ *readers if M buffers are used.*

Corollary 3: *Algorithm 1 is writer-priority if there are r+2 buffers.*

These follow from pidgeon-hole arguments.

MULTI-READER, MULTI-WRITER SOLUTIONS

In order to extend Algorithm 1 to a multi-writer solution, it is critical that writers preserve mutual exclusion between themselves *for each individual buffer*. Algorithm 2 uses a number of buffers private to each writer, with all the buffers sharing a common numbering system. If each writer has M buffers, then writer i has all buffers whose number is i modulo w. Each writer uses Algorithm 1, but tries to access only its own buffers. Readers use the same protocol as Algorithm 1. A multi-writer version of Lamport's regular register provides a robust, multi-writer regular register for the selector [Newman-Wolfe'86]. As in the single writer case, even though the selector by itself is not atomic, the handshaking with the other control variables makes the whole construction atomic.

Theorem 4: *Algorithm 2 implements a lockout- and deadlock-free, atomic, r-reader, w-writer buffer using safe bits.*

Lemma 5: *Algorithm 2 is writer-priority if w(r+2) buffers are used.*

The proofs of these are similar to those for Algorithm 1.

This multi-writer version does cost in space, however, as we now need $wM(2+r+b)$ bits. If there are r+2 buffers per writer then no writer will ever have to wait on a reader and we obtain a writer-priority solution to the *(r,w)-CRWW* problem at the expense of $w(r+2)(2+r+b)$ bits.

CONCLUSIONS

Algorithm 1 gives a spectrum of lockout-free, atomic *(r,1)-CRWW* soultions using regular bits and satisfying Courtois et al.'s requirements, bridging the gap between the space-efficient solutions with waiting and those without waiting. The spectrum of solutions gives a space-waiting tradeoff for the writer of

$$(space\text{-}1)\times(waiting)=r,$$

where space is measured in the number of buffers required, and waiting is the upper bound on the number of readers for which the writer may have to wait during a single write. The algorithm gives a writer-priority solution if *r+2* buffers are used.

A simple extension of the solution for one writer gives a solution for the atomic *(r,w)*-CRWW problem. It also guarantees strong serializability even with a regular selector register.

REFERENCES

[1] Courtois, P.J., F. Heymans and D.L. Parnas['71], "Concurrent Control with 'Readers' and 'Writers'," *CACM* 14:10, 667-668, 1971.

[2] Dijkstra, E.W.['65], "Solution of a Problem in Concurrent Programming Control," *CACM* 8:9, 569, 1965.

[3] Lamport,L.['74], "A New Solution of Dijkstra's Concurrent Programming Problem," *CACM* 17:8, 453-455, 1974.

[4] Lamport,L.['77], "Concurrent Reading and Writing," *CACM* 20:11, 806-811, 1977.

[5] Lamport,L.['85], "On Interprocess Communication," DEC Systems Research Center, SRC Report 8, Dec. 25, 1985.

[6] Newman-Wolfe,R.E.['86], "New Solutions to the CRWW Problem," *internal paper,* Computer Science Department, University of Rochester, March 1986.

[7] Peterson,G.L.['79], "Concurrency and Complexity," TR59, Computer Science Department, University of Rochester, 1979.

[8] Peterson,G.L.['81], "Concurrent Reading While Writing," TR67, Computer Scinece Department, The University of Rochester, 1981.

[9] Peterson,G.L.['83a], "Concurrent Reading While Writing," *TOPLAS* 5:1, 46-55, 1983.

[10] Peterson,G.L.['83b], "A New Solution to Lamport's Concurrent Programming Problem Using Small Shared Variables," *TOPLAS* 5:1, 56-65, 1983.

This work was supported in part by DARPA grant DACA76-85-C-0001.

```
Reader:
    Get number of current copy
    Post notice of interest in it
    Wait until coast is clear
    read copy
    Remove interest notice

Writer:
    Get number of current copy
    Get number of oldest copy
    While no unused copy has been reserved
        Count number of readers using this copy
        If used, try another copy, but not the current one
        Else try to reserve it
            Post interest notice
            Count readers using it again
            If reader is seen the second time through
                Defer to reader by removing interest notice
                Try another copy, but not the current copy
            Else I've successfully reserved a copy
    write buffer
    Change copy selector
    Remove interest notice
```

Fig.1. High-level view of the protocols for Algorithm 1.

```
jth Writer  Get number of current copy
            Get number of my oldest copy
            While no unused copy of mine has been reserved
                Count number of readers using this copy
                If used, try another of my copies, but not the current one
                Else try to reserve it
                    Post interest notice
                    Count readers using it again
                    If reader is seen the second time through
                        Defer to reader by removing interest notice
                        Try another of my copies, but not the current copy
                    Else I've successfully reserved a copy
            write buffer
            Change copy selector
            Remove interest notice
```

Fig.2. High-level view of the writer's protocol for Algorithm 2. The reader uses the protocol from Algorithm 1.

On Finding And Updating Shortest Paths Distributively

K.V.S. Ramarao and S. Venkatesan

Department of Computer Science

University of Pittsburgh

Pittsburgh, PA 15260

ABSTRACT

Construction and modification of shortest paths in networks are among the most fundamental problems encountered in distributed systems. We study the complexity aspects of these problems here. We present tight lower and upper bounds on the message complexity of several questions arising in this context.

1. Introduction

In this paper, we present several results on computing and updating shortest paths in a distributed environment. The model we consider is a network of n nodes, each having a distinct label. Each node has a set of communication links, each link connecting the given node with another node (a neighbor) of the network. The communication between nodes must go through these links. Each of the links has a non-negative weight and this weight is known to the end nodes of the link. In real life, the weight of an edge may represent the time taken to transmit a message or the congestion on the link, etc. All of the nodes execute the same algorithm. At any time during the execution of the algorithm, a node may (1) Send a message to a neighbor, (2) Receive a message from a neighbor, or (3) Do some local computation.

More formally, the network is represented by a weighted graph G=(V,E,W) where V is the set of nodes or vertices, E is the set of edges or links, and $W:E \rightarrow N$

where N is the the set of non-negative integers representing costs or weights. Let |V|=n and |E|=e. A **path** is a sequence of links (edges) and the **weight or cost of a path** is the sum of the weights of all of the links on the path. A shortest path between two nodes is a path of minimum weight (cost).

Under this framework, we consider two distributed shortest path problems: the single source shortest path problem and the all-pairs shortest path problem. In the **single source shortest path** problem, a distinguished node v_0, called the source node is specified and we are asked to find the (weight of the) shortest path between v_0 and each node v ϵ $\{V-v_0\}$ and build the Single Source Shortest Path Tree (SSSPT). We will assume that initially, the source node knows that it is the source. The **all-pairs shortest path problem** is to find the (weight of the) shortest paths between every pair of nodes. Given a tree T rooted at v_0, we define d(v) for each v ϵ V-$\{v_0\}$ to be the weight of the path (path going over the edges of T) from v_0 to v. For each v ϵ V-$\{v_0\}$, let D(v) be the weight of the shortest path from v_0 to v.

The performance of the distributed algorithms is measured using the worst case communication complexity. We shall use two communication complexities, viz. **message complexity** and **bit complexity. Message complexity** counts the total number of messages sent over the communication links. All of the messages are of size O(logn) or less. The **bit complexity** measure counts the total number of bits transmitted over the communication links. We assume that the messages always arrive correctly and message ordering is preserved (i.e. if a node v_1 sends two messages to a neighbor v_2 on the same edge, then v_2 receives both messages in the same order that v_1 sent them).

2. Outline of the results

In this section, we briefly outline the results. Section 2.1 contains the results for the single source shortest path problem. Section 2.2 deals with updating shortest paths (both single source and all-pairs). Section 2.3 deals with the sensitivity analysis of single source shortest paths and section 2.4 contains results on the verification of shortest paths. The lower bound arguments given below apply to the class of algorithms that are asynchronous. In proving the lower bounds, we assume that each of the nodes only knows the weights of the outgoing links and does not know the label of any of its neighbors.

2.1. Computing shortest paths

For computing a single source shortest path tree, $\Omega(n^2)$ is a lower bound on the number of messages in the worst case. We show this by using an adversary that forces any distributed algorithm to use at least $\frac{n(n-2)}{8}$ messages on a complete graph of n nodes. This technique is similar to that given in [4]. We then present an algorithm that is message–optimal.

To prove the lower bound, we take a complete graph G=(V,E) of n nodes. Partition V into two disjoint subsets V_1 and V_2 such that $|V_1|=|V_2|=\frac{n}{2}$. Let $v_0 \in V_1$. The weight function for the edges is as follows:

e_s=(u,v); W(e_s)=0 for all u,v $\in V_1$ and u,v $\in V_2$ and call e_s a **straight edge.**

e_c=(u,v); W(e_c)=1 for all u $\in V_1$, v $\in V_2$ and call this edge a **cross edge.**

We also use an adversary. Any algorithm that solves the single source shortest path problem will ask the adversary for an unused edge. The adversary gives the edges in a consistent manner. The weight assignment is known only to the adversary and not to the algorithm. We define an alternating cycle to be one where the edges alternate between straight and cross edges as the cycle is traversed. We need the following theorem due to [4] to prove the lower bound.

Theorem 1: In any complete graph of n vertices, there are $\frac{n(n-2)}{8}$ edge disjoint alternating cycles of length four.■

We show the lower bound by contradiction. Assume that there is an algorithm which computes the SSSPT using $o(n^2)$ messages. The tree produced by this algorithm will have a shortest path of weight 1 to all of the vertices in V_2. Since the algorithm used only $o(n^2)$ messages, only $o(n^2)$ edges will be used and consequently, there will exist an alternating cycle of length four consisting of only unused edges. Now the adversary swaps the weights of the adjacent edges (the cross edges of the cycle have a new weight of 0 and the straight edges have a new weight of 1). With this change, the adversary proves that the shortest path to a vertex in V_2 is of cost 0 and hence the SSSPT obtained by the algorithm is incorrect. The proof follows from Theorem 1. A similar argument is used for the minimum spanning tree problem in [4].

We now present a message-optimal algorithm for finding the SSSPT. This algorithm closely resembles Dijkstra's algorithm[1] for the sequential case and is conceptually simpler than Frederickson's[3] algorithm for the distributed case.

The algorithm consists of n–1 stages. Initially all of the nodes are marked as **available** (to be added to the tree). Whenever a node is added to the tree, it is marked as unavailable. At the i^{th} stage, the available node with the shortest distance from v_0 is added to the partial SSSPT. After n–1 stages, we have the required tree. At the i^{th} stage, the source node v_0 initiates the phase by sending a message to each of its descendants through the partial SSSPT built so far. Now, the node to be added to the SSSPT is to be identified. Each leaf node v, on receipt of this message, scans its list of neighbors who are not yet on the tree, computes the weight of the shortest path from v_0 to each of the neighbors passing through v, chooses that neighbor with the shortest path from v_0, marks that edge as the 'best edge' and sends the distance of the node from v_0 to its parent. Each non-leaf node finds its nearest available neighbor, compares it with the nodes it received from each of its children, chooses the node with the shortest distance from v_0, marks that edge as the 'best edge' and sends its distance to its parent. Finally, the root node receives this and decides which node, say v, is to join the tree now. It then sends a message to v along the path on the tree comprising of the best edges leading to v permitting it to join the tree. On receipt of such a message, v joins the tree, computes and stores D(v), its shortest distance from v_0 and informs all of its neighbors that it has joined the tree.

The proof of correctness of this algorithm follows from the correctness of Dijkstra's algorithm. The total number of messages used by this algorithm is $O(n^2)$ as it uses O(n) messages for each stage and there are n–1 stages.

2.2. Updating shortest paths

A lower bound on the bit complexity of updating the single source shortest path tree when a routing table is maintained at each node is $\Omega(n^2)$. Our proof essentially shows that updating the routing table alone takes at least $\frac{n(n-1)}{2}$ bits and hence the lower bound for updating the solution.

The routing table t in any node v is an array where t(v,u) is the child of v such that u can be found in the subtree rooted at u. Such a routing table is necessary if the shortest path tree is to be used to send some message from the source node to an arbitrary node in the network incurring the minimum cost.

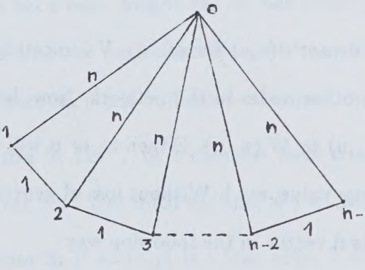


Fig 1(a)

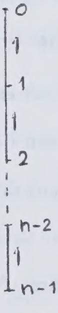


Fig 1(b)

Consider the example in Fig.1(a). The shortest path tree consists of a star network with the source node 0 as the center. Now, when the weight of the edge (0–1) changes from n to 1, the resulting shortest path tree is linear as shown in Fig 1(b). Now to update the routing table, at least one bit of information needs to be sent from the node labeled n–1 to all of the nodes in the network, and at least one bit of information needs to be sent from the node labeled n–2 to all of the nodes except n–1, and so on. This means that $O(n^2)$ bits of information will have to be sent if the routing table is to be updated correctly and thus we have the lower bound. When a routing table is not maintained, establishing a lower bound seems to be hard since there is a spectrum of possibilities for the amount of routing information stored at each node. We are currently investigating these more general cases.

We also present an algorithm that uses $O(n^2)$ messages for updating the solution to the **all-pairs shortest path problem** when the weight of one of the edges decreases. In fact, our algorithm takes only $O(n^2)$ messages even when all of the edges incident on any one node have new weights lower than their respective previous weights. We also show that when a new link is added to the network or when a new node is added to the network (along with new links incident on this node), the solution to the all–pairs shortest path problem can be updated using only $O(n^2)$ messages. On the other hand, if there are arbitrary changes, then our algorithm does not work and indeed, from the experiences in designing sequential algorithms for this case, it seems likely that the original algorithm has to be rerun.

We will assume that each node v has a vector d(v,u) for all $u \in V$ indicating the shortest distance between itself and all of the other nodes in the network. Now, let the weight of the edge (v_0,u) decrease from $W(v_0,u)$ to $W'(v_0,u)$. Either v_0 or u will start first (they can decide on this by comparing some value, etc.). Without loss of generality, assume that v_0 starts first. Now, v_0 updates its d vector in the following way:

$d'(v_0,v)=\min\{d(v_0,v), W'(v_0,u)+d(u,v)\}$................. (Eqn 1).

The node v_0 obtains d(u,v) by requesting it from its neighbor u. The node v_0 then broadcasts the new vector d' containing n entries to every node in the network using $O(n^2)$ messages. All of the other nodes update their d vectors in the following way:

$d'(v,v_0)=\min\{d(v,v_0),d(v,u)+W'(u,v_0)\}$

$d'(v,w)=\min\{d(v,w),d'(v,v_0)+d'(v_0,w)\}$ for all $w \in V-\{v_0 \cup v\}$

Each node v can update its d vector easily as the terms in the right hand sides of the above expressions are readily available to it.

A small modification to the above method yields an $O(n^2)$ message based algorithm even when all of the edges incident on a node decrease their respective weights. Let all of the edges incident on v_0 have a reduction in their respective weights. Let the degree of v_0 be k. In the above method, equation 1 is rewritten as follows:

$d'(v_0,v)=\min\{d(v_0,v),W'(v_0,u_1)+d(u_1,v), \ldots, W'(v_0,u_k)+d(u_k,v)\}$

where $u_1, \ldots, u_k$ are the nodes adjacent to v in the network. It may be be noted that equation 1 will now use at most $O(n^2)$ messages as $v \in V$ and v can take at most n different values with $k \leqslant n-1$. Thus, an edge addition (weight decreasing from infinity to the given value) or a node addition (adding k edges) can be handled using only $O(n^2)$ messages.

2.3. Sensitivity analysis

Given a shortest path tree T, the sensitivity analysis problem is to find out by how much each edge weight can be perturbed individually without changing the minimality of the shortest path tree. We show that for the sensitivity analysis, $O(n^2)$ messages are both necessary and sufficient. We need the following results from [5] for this purpose.

Lemma 2: Let T be a shortest path tree. If f=(v,w) is a non-tree edge, T remains a shortest path tree until the cost of f is decreased by more than W(f)+d(v)-d(w).■

Lemma 3: If e=(x,y) is a tree edge, T remains a shortest path tree iff the cost of e is not increased by more than W(f)+d(v)-d(w) where f=(v,w) is a non-tree (cross) edge such that w but not v is a descendant of y in T, which minimizes W(f)+d(v)-d(w).■

We now construct a network where the sensitivity analysis for the tree edges takes $\Omega(n^2)$ messages.

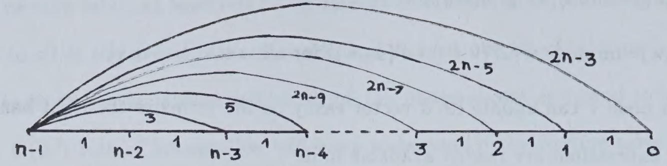


Fig 2

Consider the network given in figure 2. For the edge (n–1,n–2), the edge (n–1,n–3) is the cross edge that determines the increase in weight. Similarly, for the edge (n–2,n–3), the cross edge is (n–1,n–4) and for (n–3,n–4), the cross edge is (n–1,n–5) that determines the increase in weight. Now, any algorithm will have to use O(n) messages to compute the increase in weight for such a tree edge. As there are O(n) such edges, we need $O(n^2)$ messages.

We now give an $O(n^2)$ algorithm for the sensitivity analysis. Each node v computes d(v), the distance from v_0. This is sufficient to compute the changes in weights for the non-tree edges. For tree edges, we distribute the topology of the shortest path tree to all of the nodes using $O(n^2)$ messages. Now, each tree edge (x,y) sends a message to all of its descendents (specifying x and y) informing them of its intention to compute the allowable perturbation in its weight. Each descendant w examines each of its outgoing edges e to check that the other node v is not a descendant of y and calculates W(e)+d(v)–d(w). Now it chooses the minimum of these values and the values it got from all of its children and sends the minimum value to its parent. Clearly, O(n) messages are sufficient for each tree edge and there are n–1 tree edges and hence $O(n^2)$ messages are sufficient to do the sensitivity analysis.

2.4. Verifying shortest path trees

In this problem, each node knows its parent and its children in a tree and we are asked to verify if the given tree is indeed a single source shortest path tree. A node without a parent is the source node. We will assume that there is only one source node. For this problem, we show that O(e) messages are both necessary and sufficient.

We show that O(e) messages are necessary by contradiction. Assume that there is an algorithm that verifies the SSSPT using o(e) messages. This implies that there exists at least one edge ever which no message was sent. Without loss of generality, assume that the end nodes of this edge have different d values. Introduce a new node in between by splitting the edge. Let the weights of the edges to the new node be the same as the weight of the edge that was split. Now, make the node with higher d value, the parent of the new node. Obviously this tree is not a SSSPT, but the algorithm will verify this tree to be correct and hence o(e) messages are not sufficient.

We now present an algorithm for verifying SSSPT using O(e) messages. Given a tree T, each node v computes d(v) using n–1 messages. Now, all of the nodes exchange the d values with their neighbors using O(e) messages.

```
BEGIN
  compute d(v)  (* all of the nodes compute this using n–1 messages
  send d(v) to each neighbor u
  receive d(u) from each neighbor u and store it locally
  tree_ok[v] ← TRUE;
  FOR each u that is a neighbor of v DO
    IF d(u)+W(u,v) < d(v) THEN
      tree_ok(v) ← FALSE;
  IF v has no child THEN
    send tree_ok(v) to the parent
  ELSE
    BEGIN
      wait for a message containing tree_ok(u) from each neighbor u of v
      FOR each u, a neighbor of v, DO
        tree_ok(v) ← tree_ok(v) Λ tree_ok(u)
      send tree_ok(v) to the parent
      IF v=$v_0$ THEN (* it is the source node *)
      send tree_ok($v_0$) to all of the nodes in the network
    END
END
```

Fig 3

In the next step, each node verifies that the tree is locally correct by checking that it does not get a lower value of d by choosing some other node as its parent. If the node is a leaf node it sends the outcome of its verification to its parent. If it is an interior node, it waits for the outcomes from each of its children. Depending on its outcome and the outcomes it receives from its children, it sends the outcome of the entire subtree rooted at this node to its parent. The root node awaits the results from all of its children and makes the right choice. It then sends the final result to all of the nodes in the network. The complete algorithm is given in Fig 3.

A formal description of the other algorithms given in this paper can be obtained easily from the informal descriptions and hence we omit them here.

3. Conclusions

While we have solved several problems in distributed shortest paths, we leave some problems open. For example, the problem of updating the SSSPT when the routing table is not maintained is left open. The problem of updating the solution to the all-pairs shortest path problem when the edge weights increase is also left open. It would be interesting to see if a synchronous communication network can reduce the complexity of these problems. Also, we have not considered the time complexity of the algorithms.

4. References

[1] Dijkstra, E.W. A note on two problems in connection with graphs, Numerische Mathematik 1(1959) pp 269–271

[2] Fujishige, S, A note on the problem of updating shortest paths, Networks 11(1981), pp 317–319

[3] Frederickson, G.N., A single source shortest path algorithm for a planar distributed network, 2nd Annual Symposium on Theoretical Aspects of Computer Science, 1985, pp 143–150

[4] Korach, E., Moran, S., and Zaks, S., Finding a minimum spanning tree can be harder than finding a spanning tree in distributed networks, Technical Report #294, 1984, Department of Computer Science, Technion, Israel

[5] Tarjan, R.E., Sensitivity analysis of minimum spanning trees and shortest path trees, Information Processing Letters 14,1 (1982), pp 30–33

OPTIMAL SNAPSHOTS AND THE MAXIMUM FLOW IN PRECEDENCE GRAPHS

ELI GAFNI† and VALMIR C. BARBOSA† ‡
Computer Science Department
University of California
Los Angeles, CA 90024

ABSTRACT

We consider a distributed asynchronous system in which each processor-state is associated with a value, and each system-state is associated with the sum of the values of the constituent processor-states. A run of the system reveals a set of consistent system-states (or snapshots). These are the system-states that appear in all the total-order extensions of the partial-order of events implied by the run. Given a run we are to identify a consistent system-state of maximum (or minimum) value. This problem arises in instances of Stochastic Relaxation, like Simulated Annealing, and Evidential Reasoning over Bayes Nets. In this paper we transform the problem into a maximum-flow problem over a network which is closely related to the graph of the partial-order on the events of the run. Furthermore, a particular scheduling mechanism especially suitable for Stochastic Relaxation gives rise to runs that can be processed by breaking them into segments and processing each segment in isolation. This lends the problem to be solved "on-the-fly" while the run is in progress.

1. INTRODUCTION AND MOTIVATION

Let $X=(X_1,...,X_n)$ be a vector and let $f:R^n \rightarrow R^+$ be a positive function. Starting with an instantiation x^0 of X, consider the following probabilistic process:

Given x^k, we set $x^{k+1}:=\hat{x}$, where $\hat{x}$ is obtained by choosing an arbitrary index i, and randomly selecting a value $\bar{x}$ for X_i. The random selection is governed by the probability distribution $P(X_i=\bar{x})=\alpha_{i,k} f(y)$, where $\alpha_{i,k}$ is a normalization constant, and $y=(x_1{}^k,\ldots,x_{i-1}{}^k,\bar{x},x_{i+1}{}^k,\ldots,x_n{}^k)$. That is, $\hat{x}$ and x^k might differ only in the i^{th} entry. We then set $\hat{x}=y$.

It was essentially proved by [Gema84], that if each index i, $1 \le i \le n$, is chosen infinitely often in the probabilistic process above, then the distribution of X converges to

$$P(X=x)=\alpha f(x),$$

where α is a normalization constant.

---

† Supported by NSF grant DCR 84-51396 and by XEROX grant W850813.
‡ Supported by CAPES 3638/81-4, Ministry of Education, Brazil.

The case of interest for distributed systems is when $f(x)$ is of the special form

$$f(x)=\prod_{j=1}^{l} g_j(x),$$

where l is some positive constant and $g_j(x)$, $1 \le j \le l$, is a function that depends only on few entries of the vector X. In this case, in order to carry out the probabilistic step with respect to entry i, it is sufficient to evaluate only these g_j's that depend on X_i. All the other functions are divided out. Hopefully, since each of the functions depends only on few entries, the probabilistic step may depend only on few entries of the vector X.

This suggests immediately the possibility of parallelizing the probabilistic process. The probabilistic step with respect to two entries i and j can be carried out concurrently, provided no function $g_j(x)$, $1 \le j \le l$ depends both on X_i and X_j. Concurrent execution of these two steps is easily seen to be computationally equivalent to a serial execution of one step after the other.

An architecture that takes advantage of this parallelism consists of a network of processors P_i, $1 \le i \le n$. Each processor P_i is responsible for carrying out the probabilistic step with respect to entry i. We connect P_i and P_j by a bidirectional communication link if and only if the probabilistic step with respect to X_i and X_j cannot be carried out concurrently. In this network, independent sets of processors can carry the probabilistic step concurrently, and each processor is to be aware only of the current values of entries that correspond to its neighbors in the network. A mechanism to insure that no two neighbors carry out the probabilistic step simultaneously, and that each entry is involved in a step infinitely often, is described in this paper.

Variants of the probabilistic process outlined above occur in distributed Simulated Annealing [Kirk83] and Evidential Reasoning [Pear86]. In both cases a value of interest is x^* that minimizes (or maximizes) $f(x)$. In general, in the absence of any simplifying properties of $f(x)$ this is a difficult problem. Thus, one may settle for an approximation, by identifying the instantiation of X in the probabilistic process that minimizes (or maximizes) f over all instantiations that occurred in the run. For a sequential implementation of the probabilistic process this is quite simple. At time k one retains the minimum (or maximum) of the function seen up to then, and upon determination of x^{k+1} one evaluates $f(x^{k+1})$ and updates the minimum (or maximum).

Although the parallel execution is correct as a result of its equivalence to a serial one, two problems arise in trying to import the sequential mechanism to the parallel setting. First, the parallel execution is equivalent to many serial executions. Second, in the parallel execution no processor is aware of the value of $f(x)$ at any point of time. Each processor is only aware of the multiplicative factor (or additive amount) by which it increases or decreases (the logarithm of) $f(x)$ each time its variable changes.

In this paper we propose to identify the instantiation of X that minimizes (or maximizes) f over all the serial runs that are equivalent to the parallel run. In Section 2 we abstract the problem, pose it formally, and propose solutions via min- and max-flow techniques. In Section 3 we explain the ramification of a particular scheduling proposed in [Barb86], and show that as a result, "on-the-fly" solution is possible.

2. PROBLEM FORMULATION AND SOLUTION

A partial-order E on events of a distributed system is the transitive closure of total orders E_i on the events that occurred at P_i, $i=1,...,n$, and of a relation $<_m$ on events at different processors. Without loss of generality we assume that the sequence of events $e_{i,j}$, $j=0,1,...,j_i$ that occurred at each given P_i, $i=1,...,n$, are totally ordered according to the second index. The relation $<_m$ between events that occurred at different processors, is such that $e_{i,j}<_m e_{p,q}$ if and only if the reception of a message that was sent as direct result of event $e_{i,j}$ triggered the event $e_{p,q}$.

Let Θ denote the set of total orders which are extensions of E. Let $v_{i,j}$ be a value associated with event $e_{i,j}$, $i=1,..,n$ $j=0,1,\ldots,j_i$. Also, let $<_T$ be the relation between events corresponding to the total order T. Problem 1 is to find an index $\overline{i,j}$ that solves

$$\underset{\substack{\overline{i,j}\\ T\varepsilon\Theta}}{MIN} \quad \sum_{i,j:\, e_{i,j}<=_T e_{\overline{i,j}}} v_{i,j}.$$

To see how Problem 1 is associated with the problem outlined in the Introduction, let $v_{i,j}$ be the logarithm of the factor by which f is changed in the j^{th} time P_i carries out the probabilistic step. Had the process been carried out serially according to the order T, then at the time that event $e_{\overline{i,j}}$ is executed, the sum in Problem 1 is the logarithm of the factor by which the initial value of f differ from its current value.

Transformation of Problem 1 into an Independent-Set Problem

Without loss of generality, by possibly adding initial events of zero value, we assume that the sum in Problem 1 includes at least one value from each processor. We now relax E into E' by eliminating from E the relations between events that are solely related by "message-relation" $<_m$. Formally, E' is the transitive-closure of E_i, $i=1,...,n$, and the relation $<_{m'}$ between events that occurred at different processors, where $e_{i,j}<_{m'}e_{p,q}$ if and only if $e_{i,j+1}<_m e_{p,q}$. From the formal definition of E' it is obvious that E' is a relaxation of E. That is, every relation in E' is a relation in E.

Pictorially, Figure 1 depicts the graph whose transitive-closure captures E. This graph is a directed acyclic graph on the set of nodes that correspond to the events of the run. The partial order E' can be obtained by considering relations among arcs in Figure 1, where event $e_{i,j}$ is associated with the arc leading from node $e_{i,j}$ to node $e_{i,j+1}$, and one arc precedes another if the former's head node precedes the latter tail node.

Now, given a total order $T\varepsilon\Theta$ and an index $\overline{i,j}$, the set A of events that precede and include $e_{\overline{i,j}}$ in T, has the property that no event in $\overline{A}$, the complement of A, precedes (in E) an event in A. Similarly, given a set A with this property there exist a total-order T and and index i,j such that A is the set of events that precedes and includes $e_{i,j}$.

This property of A implies that it contains an independent set I of n events in E', one from each processor, such that the set of events in E' that precedes and includes I is A. The set I is obtained by choosing the "latest" event in A that occurred at each P_i, i=0,...,n. This set of events defines a *snapshot* - the collection of states in which each processor is left after executing the corresponding event. To see that this is an independent set, notice that if $e_{i,j}<_{E'}e_{p,q}$ and both are members of I, then $e_{i,j+1}<_E e_{p,q}$ and $e_{i,j+1}$ does not belong to A, which means that $e_{i,j+1}>_T e_{p,q}$, contradicting the requirement that T is an extension of E. Thus, with each set A we identify an independent set I in E'.

Similarly, with each independent set I in E' of cardinality n we can associate a set A of the type above by considering the set I and all the events preceding it in E'. Let $e_{i,j}$ and $e_{p,q}$ belong to I. If there is an event $e_{i,j'}$ in $\bar{A}$, preceding an event $e_{p,q'}$ in A, then $j'\geq j+1$, and $q'\leq q$, and as a result $e_{i,j}$ precedes $e_{p,q}$ in E'. This is a contradiction to the independence of the set I. Thus, with each independent set in E' we identify a set of indices for the sum in Problem 1.

As a result, we can reformulate Problem 1 in terms of independent sets in E'. With each event $e_{i,j}$ we a associate a new value $v'_{i,j}$ as follows:

$$v'_{i,j}=\sum_{l=0}^{j} v'_{i,l}.$$

Problem 2 is to solve

$$\underset{I\ \text{independent in}\ E'}{MIN} \quad \sum_{i,j:\,e_{i,j}\in I} v'_{i,j}\,.$$

We have established that Problem 1 and Problem 2 are equivalent.

Solution of Problem 2 via Min-Flow

Notice that by adding a constant to all values $v'_{i,j}$ of events that occurred at P_i we are not altering the independent set that solves Problem 2, since I includes a single event from each processor. Thus we can consider Problem 2 as a maximization problem with positive event-values, by replacing each value with its negative, and then adding a constant large enough to make all values positive. Thus w.l.o.g Problem 2 is a maximization problem, and all $v'_{i,j}$ are positive.

We now make the graph of Figure 1 into an (s,t) capacitated graph G', depicted in Figure 2, on which we will consider a min-flow problem [Even79]. With each arc from $e_{i,j}$ to $e_{i,j+1}$ we associate a lower bound $b_{i,j}=v'_{i,j}$, and an upper bound $c_{i,j}=\infty$. With each message arc we associate lower bound of zero and upper bound of infinity. We connect a new node s to all nodes $e_{i,0}$, $i=1,\ldots,n$, and connect each node e_{i,j_i}, $i=1,\ldots,n$, to a new node t. All the new arcs have lower bound of zero and upper bound of infinity. We now consider the minimum (s,t) flow in G'.

Proposition 1. The minimum (s,t) flow in G' is the value of the solution to Problem 2. Moreover, the arcs in the maximum (t,s) cut of G' is the independent set in E' that solves 2.

It is enough to prove the following Lemma. The rest follows directly from the min-flow max-cut theorem [Even79, Lawl76].

Lemma 1. There is a one-to-one correspondence between finite (t,s) cuts and independent sets in E'.

Proof. Let $(S,\overline{S})$ be a partition of the node set of G' such that S includes s, and $\overline{S}$ includes t. Denote the set of directed arcs from S to $\overline{S}$ and from $\overline{S}$ to S by $(S:\overline{S})$ and $(\overline{S}:S)$, respectively. The value of a (t,s) cut defined by a partition $(S,\overline{S})$ is the sum of the upper bounds of arcs in $(\overline{S}:S)$, minus the lower bounds of arcs in $(S:\overline{S})$. Thus in order to have a finite cut, the set $(\overline{S}:S)$ must be empty. Consider arcs in $(S:\overline{S})$. Since for each processor P_i the sequence of node $s, e_{i,0}, e_{i,1}, \ldots, e_{i,j_i}, t$ is a directed path in G', there must be an arc leading from $e_{i,j}$ to $e_{i,j+1}$ such that node $e_{i,j}$ belongs to S and node $e_{i,j+1}$ belongs to $\overline{S}$ (we tacitly allow the first of the two to be s, and the last to be t). Consider all the arcs of this type. Suppose that they do no constitute an independent set of arcs in G'. Let $e_{i,j}$ and $e_{p,q}$ be in the set and $e_{i,j} <_E e_{p,q}$. Then there exists a directed path $e_{i,j}, e_{i,j+1}, \ldots, pe_{p,q}, e_{p,q+1}$, where according to our hypothesis $e_{i,j+1} \varepsilon \overline{S}$ and $e_{p,q} \varepsilon S$. But this implies that somewhere along the path there is an arc that belongs to $(\overline{S}:S)$, contradicting the finiteness of the cut.

Thus a finite (t,s) cut identifies an independent set of events-arcs in G'. The partial order implied by G' is the same as E' only that it is enlarged to include the set of new arcs, added in going from G to G'. The assumption on the positivity of the lower bounds, will preclude them from participation in the optimal cut. On the rest of the arcs, G' and E' agree, and we have identified an independent set in E'.

The reverse direction, namely that a set of arcs, one from each processor, such that the events corresponding to the arcs are independent in E', defines a finite cut in G' is left to the reader. ■

Solution of Problem 2 via Max-Flow

As before, we assume that all $v'_{i,j}$ are positive, but in contrast we consider the minimization problem. We build a graph G' as before only that "message-arcs" are of reversed directionality, and in addition we also add arcs $e_{i,j+1}$ to $e_{i,j}$. We use the following capacity assingment. All lower bounds are zero. The capacity of arc $e_{i,j}$ to $e_{i,j+1}$ is $v'_{i,j}$. All other capacities are infinity. This construction is depicted in Figure 3. We have the following proposition.

Proposition 2. The maximum (s,t) flow in G', is the value of the solution to Problem 2. Moreover, the forward arcs in the min-cut correspond to the independent set of events that solves Problem 2.

A Lemma and an argument similar to the min-flow case holds here. Essentially it boils down to showing that if all $(S:\overline{S})$ arcs are of finite capacity then the events that correspond to the arcs in $(S:\overline{S})$ are independent in E'. If they are not, we again consider the sequence $e_{i,j+1},...,e_{p,q}$ in E'. Along this sequence there must be a switch from $\overline{S}$ to S. If this switch corresponds to a message, then in G' since "message-arcs" have been reversed we have identified an infinite capacity arc in $(S,\overline{S})$. Similarly, if the switch is an "event-arc", by adding infinite capacity arcs in the backward direction in G', we have identified again an infinite capacity arc in $(S,\overline{S})$, and the result follows.

3. SCHEDULING BY EDGE REVERSAL

In [Barb86] a distributed scheduling mechanism especially suitable for Stochastic Relaxation is proposed. The mechanism, called *edge-reversal*, is based on [Gafn81, Chan84]. Basically it work as follows. Initially all the bidirectional communication links in the network that implements the relaxation are assigned a conceptual direction in such a manner that the underlying graph is acyclic. Each processor which is a sink in this conceptual graph carries out the probabilistic step. It then reverses the direction of all its adjacent links. It implements the reversal by sending its new value to its neighbors, whereby the neighbors interpret the direction of the link as incoming. The reversal of the links create new sinks, etc, and the process continues *ad infinitum*.

It is easy to see that the edge reversal guarantees no concurrent activation of neighbors, and it is free of deadlock and starvation [Chan84].

In a partial-order of events that is produced by a run that is governed by edge-reversal, the events that occurred at P_i can be partitioned into two sets of events. The set C_i, of events of reception of values from neighbors that transform P_i into a sink, and the set $\overline{C}_i$ of the remaining events. Let $e_{i,j}$ and $e_{i,m}$, $m>j$ be two consecutive events in the total order induced by C_i. Obviously we have $v'_{i,l}=v'_{i,j}$, $j \le l < m$. Moreover, given an event $e_{p,q}$, let $\overline{e}_{p,q}$ be the first event in C_p which precedes or is $e_{p,q}$. It is not difficult to see that if a set $\{ e_{i,j(i)} \}$, $i=1,...,n$, is an independent set in E', then the set $\{\overline{e}_{i,j(i)}\}$, $i=1,...,n$, is independent in E'. Since these two sets have the same value in Problem 2, we can w.l.o.g restrict E to be the partial order induced by the union of events in C_i, $i=1,...,n$.

We now notice that if P_i and P_p are neighbors, then E restricted to the events that occurred at P_i and P_p is a total-order. Moreover, in this total-order the events at P_i alternate with the events at P_p. As a result, if $e_{i,j}$ and $e_{p,q}$ are independent in E', then $|j-q| \le 1$. Inductively, since we assume that the network is connected, we have that if $e_{i,j}$ and $e_{\hat{e},\hat{j}}$ are independent in E' then $|j-\hat{j}| \le n$. Thus, the elements of an independent set are not "far" apart.

This suggests the following way of processing the partial-order: We first compute $v'_{i,j}$ for all i,j, and then consider a sequence of partial-orders, $Pr_1, Pr_2, \ldots$. The partial order Pr_k is the one induced by the events $e_{i,j}$, $i=1,\ldots,n$, $(k-1)n \le j \le (k+1)n$, i.e. it is a "slice" of length $2|p|+1$ of the original partial-order. For each partial-order we solve Problem 2. The minimum or maximum (depending whether one solves minimization or maximization problem) of all these solution is the solution to the original problem. The reason being that the subproblems overlap sufficiently so that every independent set resides entirely in at least one of the subproblems.

To implement the partial-order processing on-the-fly, we appoint a "leader" who collects (via messages) all the $v'_{i,j}$ as the corresponding events occur. By the knowledge of the initial conceptual orientation of links, the leader can process Pr_1 once it receives all $v'_{i,j}$, $i=1,\ldots,n$, $j=0,\ldots,2n$, etc.

Various alternatives by which Pr_i is processed by distributed max-flow [Awer85, Gold86, Marb84] are also possible. The point is that in both cases one can discard history as the probabilistic process progresses.

We mention in passing that when one transforms the problem into a max-flow problem, one can discard the backward arcs $e_{i,j+1}$ to $e_{i,j}$ since the reversal of "message-edges" establishes an infinite capacity path from the first node to the second.

4. CONCLUSION

We have shown the relation between snapshots and cut-sets. Finally, for a particular application and scheduling we have proposed on-the-fly solution. A certificate to the practicality of the problem solved in this paper is that although most people "feel" what the problem is, it is rather difficult to formulate it without the precise notion of a snapshot [Chan85]. This notion was made precise only recently. If it was not around it would have to be invented just for this problem.

A point of interest is that by considering all total-order extensions of the run instead of a single one, we have exponentially (in n) more instantiations of X. The improvement to the approximation that this gets us is questionable, since the instantiations are not uniformly spread over the problem domain, but are rather very much correlated. We are now experimenting with various examples in order to be able to quantify this improvement.

REFERENCES

[Awer85] Awerbuch, B., ''Complexity of Network Synchronization,'' *Journal of the ACM* **32**(4), pp.804-823 (October 1985).

[Barb86] Barbosa, V.C., ''Concurrency in Systems with Neighborhood Constraints,'' Ph.D. Dissertation, Computer Science Dept., University of California, Los Angeles, CA (1986).

[Chan84] Chandy, K. M. and J. Misra, ''The Drinking Philosophers Problem,'' *ACM Transactions on Programming Languages and Systems* **6**(4), pp.632-646 (October 1984).

[Chan85] Chandy, K. M. and L. Lamport, ''Distributed Snapshots: Determining Global States of Distributed Systems,'' *ACM TOCS* **3**(1), pp.63-75 (February 1985).

[Even79] Even, S., *Graph Algorithms,* Computer Science Press, Rockville, Maryland (1979).

[Gafn81] Gafni, E. M. and D. P. Bertsekas, ''Distributed Algorithms for Generating Loop-Free Routes in Networks with Frequently Changing Topology,'' *IEEE Transactions on Communications* **COM-29**(1), pp.11-18 (January 1981).

[Gema84] Geman, S. and D. Geman, ''Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence* **PAMI-6**(6), pp.721-741 (Nov. 1984).

[Gold86] Goldberg, A. V. and R. E. Tarjan, ''A New Approach to the Maximum Flow Problem,'' pp. 136-146 in *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, Berkeley, CA (1986).

[Kirk83] Kirkpatrick, S., C. D. Gelatt, Jr., and M. P. Vecchi, ''Optimization by Simulated Annealing,'' *Science* **220**(4598), pp.671-680 (May 1983).

[Lawl76] Lawler, E. L., *Combinatorial Optimization: Networks and Matroids,* Holt, Rinehart and Winston, New York, NY (1976).

[Marb84] Marberg, J.M. and E.M. Gafni, ''An $O(N^3)$ Distributed Max-Flow Algorithm,'' pp. 478-482 in *Proceedings 1984 Conf. on Information Sciences and Systems*, Princeton University, Princeton, NJ (1984).

[Pear86] Pearl, J., ''Evidential Reasoning Using Stochastic Simulation of Casual Models,'' R-68-I, Computer Science Dept., University of California, Los Angeles, CA (October 1986).

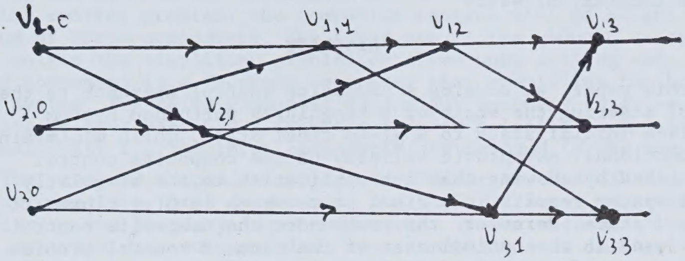

V2,0
V1,1
V12
V13
V2,0
V2,1
V2,3
V3,0
V31
V33

Fig 1

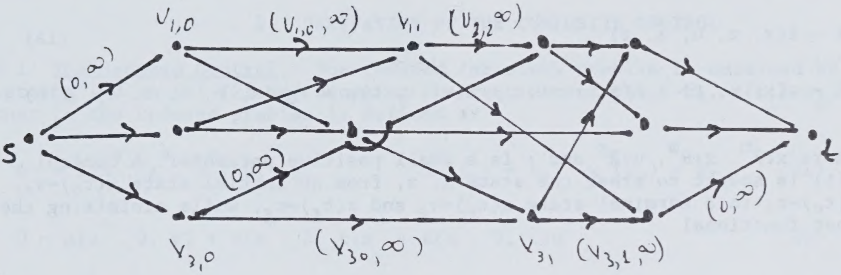

V1,0
(V1,0,∞)
V1,1
(V2,2∞)
(0,∞)
S
(0,∞)
t
(0,∞)
V3,0
(V30,∞)
V31
(V3,1,∞)

Fig 2

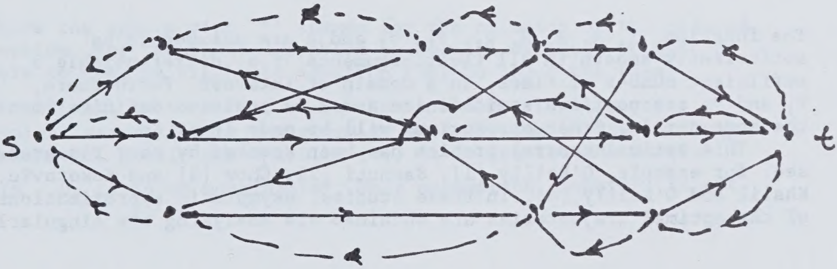


Fig 3

COMPOSITE CONTROL OF SINGULARLY PERTURBED SYSTEMS WITH APPLICATION TO FLIGHT MECHANICS PROBLEMS

Y.-N. HU and H.K. KHALIL
Department of Electrical Engineering and Systems Science
Michigan State University
East Lansing, MI 48824

ABSTRACT

In this paper, we develop a composite control approach to the problem of steering the state of a singularly perturbed system from a given initial state to a given final state, while minimizing a cost functional. Asymptotic validity of the composite control is established by showing that its application to the singularly perturbed system results in a final state which is $O(\epsilon)$ closed to the desired state. Moreover, the cost under the composite control is $O(\epsilon)$ closed to the optimal cost of the reduced control problem. The performance of the composite control is illustrated by application to the minimum time-to-climb problem in flight mechanics.

1. INTRODUCTION

Consider the singularly perturbed system

$$\dot{x} = f(x, z, u, \epsilon, t) \tag{1a}$$

$$\epsilon\dot{x} = a(x, \epsilon, t) + A(x, \epsilon, t)z + B(x, \epsilon, t)u + \epsilon g_1(x, z, u, \epsilon, t) \tag{1b}$$

where $x\epsilon R^n$, $z\epsilon R^m$, $u\epsilon R^r$ and ϵ is a small positive parameter. A control $u(t)$ is sought to steer the state x, z, from an initial state $x(t_0)=x_0$, $z(t_0)=z_0$ to a terminal state $x(t_f)=x_f$ and $z(t_f)=z_f$, while minimizing the cost functional

$$J = \int_{t_0}^{t_f} [V_1(x, \epsilon, t) + z^T V_2(x, \epsilon, t)z + u^T R(x, \epsilon, t)u]dt \tag{2}$$

The function, f, a, A, B, g_1, V_1, V_2 and R are assumed to be sufficiently smooth in all their arguments, i.e. differentiable a sufficient number of times, in a domain of interest. Furthermore, V_1 and V_2 are positive semidefinite and R is positive definite in the same domain. Other assumptions will be made later on.

This optimal control problem has been studied by many researchers; see, for example, O'Reilly [1], Sannuti [2], Chow [3] and Kokotovic, Khalil and O'Reilly [4]. In these studies, asymptotic approximations of the optimal trajectories are obtained via analyzing the singularly perturbed two-point boundary value problem that results from applying the maximum principle. These asymptotic approximations have been extensively used in flight mechanics problems; see, for example, Kelley [5], Ardema [6] and Calise [7].

We develop a composite control approach to the steering control problem. Composite control of singularly perturbed system has been known in the context of stabilizing feedback control[4]. In our steering control problem, the composite control will be sought as the sum of three components. The first one is the reduced control which solves the simplified problem obtained upon setting $\epsilon=0$. The second component is a feedback component that stabilizes the boundary-layer system. The third component is a right boundary-layer component that steers fast variable z from the reduced solution to the desired terminal state z_f. The three components are derived in the next section.

The proposed composite control is similar to that of Chow [3] in that it comprises three components, but with different procedures of calculating the boundary-layer components. In our method, the boundary-layer controls does not optimizes cost functionals as in Chow. the left boundary-layer control is a feedback stabilizing control that ensures boundary-layer stability, while the right boundary-layer control is the well-known minimum energy control that steers the state of the linear boundary-layer system to its target state. Our analysis does not involve asymptotic analysis of the full optimal control. Therefore, our assumption are weaker than those of Chow.

2. DERIVATION OF THE COMPOSITE CONTROL

2.1 The Reduced Control: The reduced (or slow) problem is obtained by setting $\epsilon=0$ in (1)-(2) and dropping the requirement $z(t_0)=z_0$, $z(t_f)=z_f$, that is the reduced problem is defined as

$$\dot{x} = f(x^0, z^0, u^0, 0, t), \qquad x^0(t_0) = x_0, \quad x^0(t_f) = x_f \tag{3a}$$

$$0 = a(x^0, 0, t) + A(x^0, 0, t)z^0 + B(x^0, 0, t)u^0 \tag{3b}$$

$$J^0 = \int_{t_0}^{t_f} [V_1(x^0, 0, t) + (z^0)^T V_2(x^0, 0, t)z^0 + (u^0)^T R(x^0, 0, t)u^0]dt \tag{4}$$

where the superscript "^{0}" stands for the solution of the reduced problem. For the reduced problem to be well-defined, we must be able to use the algebraic equation (3b) to reduc the (n+m)-dimentional state vector x^0, z^0 to an n-dimensional vector. A typical assumption in the singular perturbation literature, e.g., Kokotovic, Khalil and O'Reilly [4], is to require the matrix $A(x^0, 0, t)$ to be nonsingular. This assumption, however, is not needed in the asymptotic analysis of the two-point boundary-layer value problem associated with the full problems that arise in flight mechanics. Therefore, we make the following weaker assumption

Assumption 1: The mx(m+r) matrix $[A(x^o, 0, t) \quad B(x^o, 0, t)]$ has m linearly independent columns in the domain of interest.

This assumption implies that rank$[A \quad B]$=m for all x^o and t, but not vice-versa. A weaker assumption would be to require rank $[A \quad B]$=m, but this would complicate the analysis. Assumption 1 guarantees the existence of a permutation matrix P such that the first m columns of $[A \quad B]P$ are linearly independent for all x^o and t, e.g.

$$[A_2(x^o, t) \quad B_2(x^o, t)] = [A(x^o, t, 0) \quad B(x^o, 0, t)]P \tag{5}$$

where $A_2(x^o, t)$ is nonsingular. Defining $\bar{z}^o$ and $\bar{u}^o$ by

$$\begin{bmatrix} z^o \\ u^o \end{bmatrix} = P \begin{bmatrix} \bar{z}^o \\ \bar{u}^o \end{bmatrix} \triangleq \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \begin{bmatrix} \bar{z}^o \\ \bar{u}^o \end{bmatrix} \tag{6}$$

the algebraic equation (3b) can be rearranged as

$$0 = a + [A \quad B]PP^T \begin{bmatrix} z^o \\ u^o \end{bmatrix} = a + [A_2 \quad B_2] \begin{bmatrix} \bar{z}^o \\ \bar{u}^o \end{bmatrix} \tag{7}$$

Hence, the reduced problem (3)-(4) can be rewritten as

$$\dot{x} = \bar{f}(x^o, \bar{z}^o, \bar{u}^o, t), \qquad x^o(t_0)=x^o, \quad x^o(t_f)=x_f \tag{8a}$$

$$0 = a(x^o, 0, t) + A_2(x^o, t)\bar{z}^o + B_2(x^o, t)\bar{u}^o \tag{8b}$$

$$J^o = \int_{t_0}^{t_f} \{V_1(x^o,0,t) + [(\bar{z}^o)^T \quad (\bar{u}^o)^T]P^T \begin{bmatrix} V_2(x^o,0,t) & 0 \\ 0 & R(x^o,0,t) \end{bmatrix} P \begin{bmatrix} \bar{z}^o \\ \bar{u}^o \end{bmatrix} \}dt \tag{9}$$

where $\bar{f}(x^o, \bar{z}^o, \bar{u}^o, t)=f(x^o, P_{11}\bar{z}^o+P_{12}\bar{u}^o, P_{21}\bar{z}^o+P_{22}\bar{u}^o, 0, t)$, and $A_2(x^o,t)$ is nonsingular. Note that if $A(x^o,0,t)$ is nonsingular to start with, the permutation matrix P can be taken as the identity matrix. Problem (8)-(9) satisfies the standard assumption of nonsingularity of $A_2(x^0,t)$. Substitution of

$$\bar{z}^0 = -A_2^{-1}(a + B_2\bar{u}^0) \tag{10}$$

into (8a) and (9) yields

$$\dot{x}^0 = \bar{f}^0(x^0, \bar{u}^0, t), \qquad x^0(t_0)=x_0, \quad x^0(t_f)=x_f \tag{11}$$

$$J^0 = \int_{t_0}^{t_f}[Q_0(x^0, t) + 2D_0^T(x^0, t)\bar{u}^0 + (\bar{u}^0)^T R_0(x^0, t)\bar{u}^0]dt \tag{12}$$

where

$$\bar{f}^0(x^0,\bar{u}^0,t)=\bar{f}(x^0,-A_2^{-1}(a+B_2\bar{u}^0),\bar{u}^0,t), \qquad Q_0=V_1+a^TA_2^{-T}M_{11}A_2^{-1}a$$

$$D_0=B_2^TA_2^{-T}M_{11}A_2^{-1}a-M_{12}^TA_2a, \qquad R_0=B_2^TA_2^{-T}M_{11}A_2^{-1}B_2-B_2^TA_2^{-T}M_{12}-M_{12}^TA_2^{-1}B_2+M_{22} \tag{13}$$

$$M_{11}=P_{11}^TV_2P_{11}+P_{21}^TRP_{21}, \quad M_{12}=P_{11}^TV_2P_{12}+P_{12}^TRP_{22}, \quad M_{12}=P_{12}^TV_2P_{12}+P_{12}^TRP_{22}$$

The reduced control problem (11), (12) is formally correct in the sense that its necessary conditions for optimality coincide with the necessary conditions of the full problem (1)-(2) upon setting $\epsilon=0$. This fact is shown in section 6.3 of Kokotovic, Khalil and O'Reilly [4] for the reduced problem (11)-(12). We assume that

Assumption 2: The reduced control problem (11)-(12) has a unique optumal solution $\bar{u}^0$, x^0, which is continuously differentiable on $[t_0,t_f]$. Once $\bar{u}^0$ and $\bar{x}^0$ are calculated, $\bar{z}^0(t)$ can be obtained from (10), and the reduced control $u^0(t)$ is given by

$$u^0 = P_{21}\bar{z}^0 + P_{22}\bar{u}^0 \tag{14}$$

2.2 Boundary-Layer Stabilizing Control: A characteristic phenomenon of singularly perturbed system is the presence of boundary layers during which the fast variable z approaches it reduced, or quasi-steady state trajectory. For this phenomenon to take place, the boundary-layer system needs to be asymptotically stable, see Kokotovic, Khalil and O'Reilly [4]. For the system (1) this will be the case if the matrix

$A(x,0,t)$ is Hurwitz uniformly in x and t, i.e., $\mathrm{Re}\lambda[A(x,0,t)]\leq -c<0$ for all x and t in the domain of interest. If this condition holds, application of the reduced control (14) to the system (1) will result in trajectories of x, z and u which approach $\overset{o}{x}$, $\overset{o}{z}$ and $\overset{o}{u}$ after a boundary-layer. If $A(x,0,t)$ is not Hurwitz, feedback must be used to stabilize the boundary-layer system. It is improtant to notice that without feedback, the boundary-layer will be unstable even when open-loop boundary-layer corrections are added to $\overset{o}{u}$. This fact was demonstrated on a second-order example in Wilde and Kokotovic [8], and on a flight mechanics model in Hu [9]. The boundary-layer system is obtained by expressing (1b) in the fast time-scale $\tau=(t-t_1)/\epsilon$, $t_1\geq t_0$, and then setting $\epsilon=0$. It is given by

$$\frac{dz}{d\tau} = a(x,\ 0,\ t) + A(x,\ 0,\ t)z + B(x,\ 0,\ t)u \tag{15}$$

where $u=\overset{o}{u}+u_F$ and x, $\overset{o}{u}$ and t are frozen at their values at $t=t_1$. Notice that the boundary-layer stability should hold along the reduced trajectory and not only at the initial time t_0. If $A(x,0,t)$ is not Hurwitz or it is Hurwitz but its stability properties are not adequate, u_F can be used to stabilize the system. Let us first shift the equilibrium of (15) to the origin. The steady-state of (15), with $u_F=0$ is

$$0 = a(x,\ 0,\ t) + A(x,\ 0,\ t)\overset{o}{z} + B(x,\ 0,\ t)\overset{o}{u} \tag{16}$$

Substracting (16) from (15) and setting $z_F=z-\overset{o}{z}$, we obtain

$$\frac{dz_F}{d\tau} = A(x,\ 0,\ t)z_F + B(x,\ 0,\ t)u_F \tag{17}$$

with x and t treated as fixed parameters. The system (17) is a linear system whose stabilizability is stated in the following assumption.
Assumption 3: The pair $[A(x,0,t)\ \ B(x,0,t)]$ is stabilizable uniformly in x and t in the domain of interest, e.g., there exists a sufficiently smooth matrix $K(x,t)$ such that

$$\mathrm{Re}\lambda(A(x,0,t) + B(x,0,t)K(x,t)) < -c < 0 \tag{18}$$

Thus, the boundary-layer stabilizing control is given by

$$u_F = K(x,t)z_F = K(x,t)(z - \overset{o}{z}) \tag{19}$$

2.3 Right Boundary-Layer Control: So far, we have derived the reduced control u^0 and the boundary-layer stabilizing control u_F. Application of $u=u^0+u_F$ to the system (1) will result in a trajectory $x(t)$, $z(t)$ that approaches the reduced state $x^0(t)$, $z^0(t)$ after an $O(\epsilon \ln l/\epsilon)$ boundary-layer and then moves along it. At the terminal time t_f, $x(t_f)$ and $z(t_f)$ will be in an $O(\epsilon)$ neighborhood of $x^0(t_f)=x_f$ and $z^0(t_f)$. Since $z^0(t_f)\neq z_f$, in general, a terminal boundary-layer control u_b should be added to u^0+u_f, whose function would be to steer $z(t)$ from an $O(\epsilon)$ neighborhood of the desired state z_f. This motion will take place over a time interval $[t_f-\Delta, t_f]$, $\Delta>0$. To derive u_b, we consider the boundary-layer system (15) with $u=u^0+u_F+u_b$ and with $t_1=t_f$, that is with x, z^0, u^0 and t frozen at $x(t_f)$, $z^0(t_f)$, $u^0(t_f)$ and t_f, respectively. Anticipating that $x(t_f)$ will be within $O(\epsilon)$ of x_f, we freeze x at x_f instead of $x(t_f)$. Thus, the right boundary-layer model is given by

$$\frac{dz}{d\sigma} = a(x_f,\ 0,\ t) + H(x_f,\ t_f)z + B(x_f,\ 0,\ t_f)[u^0(t_f) - K(x_f,\ t_f)z^0(t_f) + u_b] \tag{20}$$

where $\sigma=(t-t_f)/\epsilon$ and $H=A+KB$. Equation (20) has equilibrium at $z=z^0(t_f)$. Defining $z_b=z-z\ (t_f)$, we obtain

$$\frac{dz_b}{d\sigma} = H(x_f,\ t_f)z_b + B(x_f,\ 0,\ t)u_b \tag{21}$$

The right boundary-layer control problem is to move z_b from $z_b=0$ at $\sigma=-\Delta/\epsilon$ to $z_b=z_f-z^0(t_f)$ at $\sigma=0$. Since the system (21) is linear, the solution of this steering control problem is the well-known minimum energy control. It requires the following assumption

Assumption 4: The pair $[A(x_f,0,t)\quad B(x_f,0,t)]$ is controllable.

Since $H=A+BK$, controllability of (A,B) implies controllability of (H,B) for any matrix K. The minimum energy control that moves z_b from $z_b(-\Delta/\epsilon)=0$ to $z_b(0)=z_f-z(t_f)$ is given by

$$u_b(\sigma) = B^T(e^{-\sigma H})^T W^{-1}(\Delta/\epsilon)(z_b(0) - e^{H\Delta/\epsilon}z_b(-\Delta/\epsilon)) \tag{22}$$

where the controllability Grammian $W(\Delta/\epsilon)$ is given by

$$W(\Delta/\epsilon) = \int_{-\Delta/\epsilon}^{0} e^{-\sigma H}BB^T(e^{-\sigma H})^T d\sigma = \int_0^{\Delta/\epsilon} e^{\lambda H}BB^T(e^{\lambda H})^T d\lambda \tag{23}$$

In (22) and (23) the arguments of $H(x_f,t_f)$ and $B(x_f,0,t_f)$ are omitted for convenience. Due to Hurwitz property of H, the matrix $W(\Delta/\epsilon)$ may be approximated by $W_\infty = W(\infty)$, see Hu [9]. Thus, u_b can be approximated by

$$u_b(\sigma) = B^T(e^{-\sigma H})^T W^{-1}(\infty)(z_f - z^0(t_f)) \tag{24}$$

where $W(\infty)$ satisfies the algebraic Lyapunov equation. In the t time scale, u_b is given by

$$u_b(t) = B^T(e^{(t_f - t)H/\epsilon})^T W^{-1}(\infty)(z_f - z^0(t_f)) \tag{25}$$

It is importantant to notice that $u_b(t)$ is independent of the constant Δ. It dependent only on the final time t_f. If t_f is known, the expression (25) can be calculated and implemented from the initial time t_0. When t_f is free, it should be estimated.

The composite control is taken as

$$u_c = u^0(t) + K(x,\ t)[z(t) - z^0(t)] + B^T(x_f,\ 0,\ t).$$

$$[\exp(t_f - t)H(x_f,\ t_f)/\epsilon]^T W^{-1}(\infty)(z_f - z^0(t_f)) \tag{26}$$

Asymptotic validity of the composite control is established by the following theorem, whose proof can be found in [9].

THEOREM 1: Suppose that Assumptions 1-4 hold and that the composite control (26) is applied to the singularly perturbed system (1). Then, there exists $\epsilon^* > 0$ such that for all $\epsilon \in (0,\ \epsilon^*]$

$$\left\|x(t_f) - x_f\right\| = O(\epsilon),\quad \left\|z(t_f) - z_f\right\| = O(\epsilon),\quad \left\|J(u_c) - J^0(u^0)\right\| = O(\epsilon) \tag{27}$$

3. APPLICATION OF THE COMPOSITE CONTROL TO A FLIGHT MECHANICS PROBLEM

We apply the composite control strategy to the optimal maneuvers of an aircraft in a vertical plane. One of the examples of interest is the minimum time-to-climb problem.

The equations of motion for flight in a vertical plane are given by

$$\frac{dE}{dt} = \frac{V}{m}[T(E,V) - \frac{1}{2}\rho V^2 s(C_{D_0}(V) + \eta(V)C_{L\alpha}(V)\alpha^2)] = f(E,V,\gamma,\epsilon,t) \quad (28a)$$

$$\epsilon\frac{dV}{dt} = \frac{\epsilon g}{W}[T(E,V) - \frac{1}{2}\rho V^2 s(C_{D_0}(V) + \eta(V)C_{L\alpha}(V)\alpha^2)] - gSIN\gamma$$

$$= -gSIN\gamma + \epsilon g_1(E,V,\gamma,\alpha,\epsilon,t) \quad (28b)$$

$$\epsilon\frac{d\gamma}{dt} = \frac{g}{2w}\rho V s C_{L\alpha}(V)\alpha - \frac{g}{V}COS\gamma \quad (28c)$$

For definition of variables the reader is referred to [4] or [6]. A control $\alpha(t)$ is sought to steer the state $E(t_0)=E_0$, $V(t_0)=V_0$, $\gamma(t_0)=\gamma_0$ to a terminal state $E(t_f)=E_f$, $V(t_f)=V_f$, $\gamma(t_f)=$free, while minimiging t_f. The model (28) is in the singularly perturbed form but it is not in the form (1). To bring the model (28) into the form (1) we use the state transformation $Z = -gSIN\gamma$ and the feedback

$$\alpha(t) = \frac{2}{g^2\rho V s C_{L\alpha}(V)COS\gamma/W}[\frac{g^2}{V}COS^2\gamma - u] \quad (29)$$

to obtain

$$\frac{dE}{dt} = f(E,V,Z,u,\epsilon,t) \quad (30a)$$

$$\epsilon\frac{dV}{dt} = Z + \epsilon g_1(E,V,Z,u,\epsilon,t) \quad (30b)$$

$$\epsilon\frac{d\gamma}{dt} = u \quad (30c)$$

which is of the form (1). The composite control can now be derived as in the previous section.

The reduced control: The reduced problem is given by

$$\dot{E}^0 = f(E^0,V^0,0,0,0,t), \quad E^0(t_0) = E_0, \; E^0(t_f) = E_f \quad (31a)$$

$$0 = Z^0, \quad 0 = u^0 \quad (31b)$$

which is not a standard singularly perturbed form, but it satisfies Assumption 1. Hence, there exist a constant permutation matrix P such that

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}}_{P} \underbrace{\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}}_{P^T} \begin{bmatrix} V^0 \\ Z^0 \\ u^0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} u^0 \\ Z^0 \\ v^0 \end{bmatrix} = 0 \tag{32}$$

The permutation has the meaning of using the velocity V^0 as the control variable in the reduced problem.

The boundary-layer stabilizing control: In order to stabilize the boundary-layer, u_F is chosen as

$$u_F = -b_1(V - V^0) - b_2 Z , \qquad b_1 > 0,\ b_2 > 0 \tag{33}$$

The right boundary-layer control: The right boundary-layer control u_b is given by

$$u_b = B^T(e^{H(t_f - t)/\epsilon})^T W^{-1}(\infty)\left(\begin{bmatrix} Z_f \\ V_f \end{bmatrix} - \begin{bmatrix} 0 \\ V^0(t_f) \end{bmatrix}\right), \tag{34}$$

The composite control u_c is taken as

$$\alpha_c = \frac{2}{g^2 \rho V s C_{L\alpha}(V) \mathrm{COS}\gamma / W}\left[\frac{g^2}{V}\mathrm{COS}^2\gamma - (u^0 + u_F + u_b)\right] \tag{35}$$

The composite control α_c is applied to the system (31), and the resulting energy E, velocity V, and flight trajectory (path) are shown in Figs.1-3, respectively. In Fig.2, it is seen that this composite control steers the state of the aircraft to a final state (2.06 Mach and altitude 79733ft) which is about 3% error relative to the given final velocity (2.00Mach), and about .333% error relative to the given final altitude (80000ft). The total error is about 3.333% which is $O(\epsilon)$ close to the desired state. The comparision of the composite control α_C with the approximate control α_{app} which is obtained by off-line singular perturbation approximation of the optimal trajectories [6] is also shown in Fig.4.

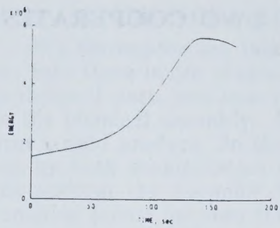


Fig 1 Energy time history for airplane 2 with composite control

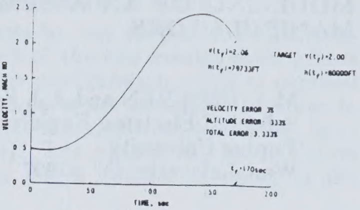


Fig.2 Velocity time history for airplane 2 with composite control

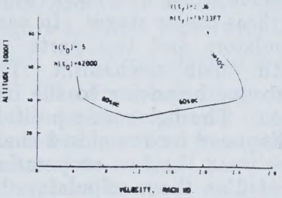


Fig 3 Flight trajectory for airplane 2 with composite control

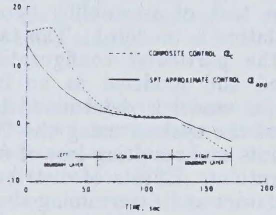


Fig 4 Comparison of the approximate control and composite control

REFERENCES

[1] R.E.O'Malley, Jr. "Introduction to Singular Perturbations," Academic Press, New York, 1974.

[2] P.Sannuti, "A Note on Obtaining Reduced Order Optimal Control Problems by Singular Perturbations," IEEE Trans. Aut. Control, AC-19, PP.256, 1974.

[3] J.H.Chow, "A Class of Singularly Perturbed Nonlinear Fixed end Point Control Problems," J. Optim. Theory, April 29, PP.231-281, 1979.

[4] P.V.Kokotovic, H.K.Khalil, and J.O'Reilly, "Singular Perturbation Methods in Control Analysis and design, " Academic Press, Chapter 1, 1986.

[5] H.J.Kelley, "Flight Path Optimization with Multiple Time-Scales," Journal of Aircraft, Vol. 8, April, 1971.

[6] M.D.Ardema, "Solution of the Minimum Time-To-Climb problem by Matched Asymptotic Expansions," AIAA Journal, Vol. 14, 1976.

[7] A.J.Calise, "Singular Perturbation Method for Variational problems in Aircraft Flight," IEEE Trans. Aut. Control, Vol. AC-21, June, 1976.

[8] R.R.Wide and P.V.Kokotovic, "Optimal Open and Closed-Open Control of Singularly Perturbed Linear Systems," IEEE Trans. Auto. Control, Vol. AC-18, PP.616-625, 1973.

[9] Y.-N.HU, "The Application of Singular Perturbation Methods to Optimal Control Problems in Flight Mechanics," Ph.D. Thesis, Dept. Elec. Eng. & Syst. Sci., Michigan State University, E. Lansing, MI. 1986.

Acknowledgement: This work has been supported in part by the National Science Foundation under Grant no. ECS-8410649.

MODELING OF AN ASSEMBLY TASK WITH TWO COOPERATING MANIPULATORS

M.A. UNSEREN and A.J. KOIVO
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907

ABSTRACT

The task of assembling two parts into a rigid product using two cooperating manipulators is modeled. The task is decomposed into three major stages. In each stage, the particular configurations of the two manipulators and two parts are identified and modeled as an open or closed kinematic chain mechanism. The dynamical model is determined by applying force and torque balancing to the free bodies of the system using the Newton-Euler formulation. The holonomic position constraints and resulting loss of degrees of freedom are discussed for the closed chain configurations. Effects of distributing a common load between the two cooperating manipulators and determining the forces and torques exerted on the manipulators by the load are discussed. A control scheme to be used in the assembly task, and particularly in the closed chain configurations, is proposed.

I. INTRODUCTION

Many benefits can be obtained by applying two cooperating manipulators to an assembly task, for example, assembling two parts into a rigid product where each part is held by a manipulator to have a coordinated assembly of the workpieces. Expensive custom-made fixturing is required for a single manipulator to perform the same task. With two cooperating manipulators, the fixturing can be eliminated and the system can be made adaptive to changes in part designs or assembly requirements. Consequently, the two cooperating manipulators can perform tasks in less time than a single manipulator. Two cooperating manipulators can perform tasks which a single manipulator is not capable of, such as transporting objects beyond the load capacity of a single manipulator. Another example is the transport of an object to a location beyond the work envelope of a single manipulator. The object can be transported to locations within the increased combined work envelope of the two cooperating manipulators by directly exchanging it between the manipulators during the transport. Two human hands perform most assembly tasks better than a single hand. The same idea applies to two cooperating manipulators in assembly.

The literature on assembly by two cooperating manipulators is quite scarce. Special problems unique to cooperating manipulators are identified in [1]. Research issues for automatic assembly are discussed in [2], in the framework of two cooperating manipulators.

II. PROBLEM FORMULATION FOR TWO COOPERATING MANIPULATORS IN ASSEMBLY

The problem is to develop a mathematical model for an assembly task performed by two cooperating manipulators. We address the problem of assembling two parts into a rigid product: (i) the manipulators grasp a respective part, (ii) cooperatively assemble them without the use of intermediate fixturing, and (iii) jointly transport the product to a specified location. A typical example of such an application is the assembly of a piston and connecting rod, the subsequent transfer of the piston/rod, and the insertion of the end product into an engine block cylinder. It is noted that such an assembly task can be decomposed into subtasks, which are discussed next.

III. THE MAJOR STAGES OF THE ASSEMBLY TASK

We decompose the task of assembling two parts by two cooperating manipulators into three major stages. In the first stage, each of the two manipulators grasps an assigned part, positions it, and orients it into the configuration which is required for the physical assembly. In the second stage, the parts are assembled together to form a rigid product. In the third stage, the product is transported to another location by both manipulators (e.g., due to its weight). The cooperating manipulators will perform the assembly task through the three stages without stopping at the transition points between the stages.

As a preliminary study, we consider two cooperating planar manipulators which perform the assembly of two parts in a vertical plane. Each manipulator has three links and three revolute joints. A typical configuration of the system is shown in Figure 1, which represents the overall system in the third stage.

III.1. The First Stage of the Assembly

The first stage of the assembly task can be considered as a preassembly operation. At the beginning of the first stage, each of the two manipulators moves from its initial position to a position which is, say, three centimeters above the center point of its assigned part, then moves down to grasp the part. Then each manipulator moves the part held by its gripper into a position and orientation so that the approach vectors of the grippers coincide. The grippers then move toward each other along their approach vectors. The first stage ends when the centers of the grippers (hands) are at a certain distance apart.

In order to control the planar manipulators to perform the first stage, dynamical models for the manipulators must be determined. The dynamical models are obtained by applying force and torque balancing to the free bodies of the system using the Newton-Euler formulation [3,4]. The general form of the equations of motion for arm #1 is given in equation (A.1) of the appendix. The external forces ${}^1f_{3,4x}$, ${}^1f_{3,4y}$ and torque ${}^1n_{3,4}$ in equation (A.1) are zero before arm #1 picks up the assigned part, but are non-zero upon holding the part. Provided we know the geometry, mass and inertia of the held part, we can determine the mathematical relations between the external forces and torques, the joint variables and link parameters by free body analysis.

As an example, consider figure 2 where arm #1 holds a load of length ℓ_4, mass m_4 and inertia I_4. The force and torque balance equations for the free body are:

$$ {}^1f_{3,4x} = m_4\ddot{x}_{4c.m.} \tag{1} $$

$$ {}^1f_{3,4y} = m_4\ddot{y}_{4c.m.} + m_4 g_y \tag{2} $$

$$ {}^1n_{3,4} = I_4\alpha_4 - r_{4x}\,{}^1f_{3,4y} + r_{4y}\,{}^1f_{3,4x} \tag{3} $$

$$ x_{4c.m.} = \ell_1\cos\theta_1 + \ell_2\cos(\theta_1+\theta_2) + (\ell_3 + \frac{1}{2}\ell_4)\cos(\theta_1+\theta_2+\theta_3) \tag{4} $$

$$ y_{4c.m.} = \ell_1\sin\theta_1 + \ell_2\sin(\theta_1+\theta_2) + (\ell_3 + \frac{1}{2}\ell_4)\sin(\theta_1+\theta_2+\theta_3) \tag{5} $$

$$ r_{4x} = \frac{1}{2}\ell_4\cos(\theta_1+\theta_2+\theta_3)\ ;\ r_{4y} = -\frac{1}{2}\ell_4\sin(\theta_1+\theta_2+\theta_3) \tag{6} $$

$$ \alpha_4 = \ddot{\theta}_1 + \ddot{\theta}_2 + \ddot{\theta}_3 \tag{7} $$

where $\vec{r}_4$ is a vector from the center of mass of the mated part to the point of application of $\vec{f}_{3,4}$. ℓ_1, ℓ_2, ℓ_3 are the link lengths of arm #1, $(\theta_1,\theta_2,\theta_3)$ are the joint angles and $(\ddot{\theta}_1,\ddot{\theta}_2,\ddot{\theta}_3)$ are the joint accelerations of arm #1.
$\vec{g} = [0,-g_y]^T$ is the gravity vector where the superscript T denotes the transpose.

The external forces and torques in final form have been obtained but are omitted due to space limitations. The mathematical model for arm #1 is obtained by substituting the equations for the external forces and torques into (A.1).

In the preassembly stage, the manipulators function independently of each other. Their motions are uncoupled; thus, the manipulators are modeled as two open kinematic chain mechanisms. Studying the degrees of freedom will provide insight into the complexity of the overall motion of the two manipulator assembly system. The degrees of freedom are determined for planar kinematic chain mechanisms by the following relation [5]:

$$\mathrm{DOF} = 3(c-1) - 2j \tag{8}$$

where DOF is the number of degrees of freedom, c is the number of links, and j is the number of joints (with or without actuators).

Each manipulator representing an open chain is mounted on a stationary pillar which is considered as a zeroth link when applying equation (8) [5]. Thus, each manipulator has 4 links and 3 joints; thus, equation (8) gives:

$$\mathrm{DOF} = 3(4-1) - 2(3) = 3.$$

Since each manipulator has three degrees of freedom, it follows that the total system consisting of the two manipulators has six degrees of freedom. Consequently, all six actuated joints in the system can be controlled independently.

III.2. The Second Stage of the Assembly

In the second stage of the assembly task performs the actual assembling of the parts. At the beginning of the second stage, the centers of the grippers are a certain distance apart. The grippers are moving toward each other along their approach vectors which are on the same line. Without stopping, the parts are brought into contact for a subsequent assembly. The second stage ends just before the parts are completely put together.

Before physical contact of the parts occurs, the configuration of the system is the same as that of the first stage. Thus, the system is modeled as two open kinematic chain mechanisms and has six degrees of freedom. Upon the initial contact of the parts, the manipulators become coupled and can no longer function independently.

The overall system consisting of the two manipulators holding the contacting parts is modeled as a single closed kinematic chain mechanism. The determination of the degrees of freedom will provide insight into the behavior of the closed chain containing the two manipulators and two parts being assembled. The degrees of freedom of the system is dependent on the specific configuration under consideration.

For the closed chain configuration of the second stage, seven links can be seen: each manipulator has three links where its 3rd link and held part are considered as one link, and a reference link which connects the manipulator bases to form the closed chain [5]. There are seven joints: three actuated joints for each manipulator, and the 7th joint resulting from modeling the parts interface as a prismatic joint. By equation (8):

$$\mathrm{DOF} = 3(7-1) - 2(7) = 4$$

Thus, the degrees of freedom of the total system in this case are reduced by two when the parts being assembled are touching each other, and being moved relative to each other.

The loss of two degrees of freedom for the closed chain configuration is caused by the kinematic coupling between the manipulators, i.e., the joint variables of the two manipulators are now coupled due to the common object. The kinematic coupling causes a constrained motion. The constraints can be expressed in the form of equations relating the joint variables [4,6]. They are referred to as holonomic position constraints and have the following form for the two planar manipulator assembly system:

$$h_i(\theta_1,\theta_2,\theta_3,\theta_1',\theta_2',\theta_3') = 0 \tag{9}$$

where $h_i(.)$ is the ith holonomic position constraint of the closed chain configuration, i=1,...,N. $\theta_1,\theta_2,\theta_3$ are the joint angles of arm #1, and $\theta_1',\theta_2',\theta_3'$ are the joint angles of arm #2.

The loss of each degree of freedom in the system corresponds to one holonomic position constraint [4,6]. Thus, N=2 in equation (9) for the closed chain configuration under discussion. The specific expressions for the two holonomic position constraints of the assembly system have been obtained, but are omitted due to space limitations.

If the joint positions are related by equation (9), the joint velocities and accelerations are also related. These expressions can be determined by taking consecutive derivatives of equations (9) with respect to time [5]. They are referred to a holonomic velocity and acceleration constraints and have the following forms respectively for the two planar manipulator assembly system:

$$\dot{h}_i(\theta_1,\theta_2,\theta_3,\theta_1',\theta_2',\theta_3') = 0 \; ; \; \ddot{h}_i(\theta_1,\theta_2,\theta_3,\theta_1',\theta_2',\theta_3') = 0 \tag{10}$$

where $\dot{h}_i(.) = \dfrac{dh_i(.)}{dt}$ is the ith holonomic velocity constraint, and $\ddot{h}_i(.) = \dfrac{d^2h_i(.)}{dt^2}$ is the ith holonomic acceleration constraint, i=1,...,N.

The two holonomic velocity and acceleration constraints for the closed chain configuration under consideration are not given here explicitly due to space limitations.

The manipulators are dynamically coupled whenever the held parts to be mated come into contact. The dynamic coupling is due to transmitted forces and torques propagating through the common load between the manipulators; thus, the force and torques exerted by each manipulator on the common load are not independent. This dependence enables us to assume how the common load is distributed between the manipulators, i.e., we assume that the fractions of the total forces and torques exerted by the common load on each manipulator are specified. These fractions depend on the sizes and load capacities of the manipulators and the orientation of the common load in the vertical plane. Given these fractions and the mass, length and inertia of the common load, the exerted forces and torques can be related to the joint variables and incorporated into the dynamical model of the closed kinematic chain mechanism. Thus, the mathematical model for the closed chain configuration of the second stage is obtained.

III.3. The Third Stage of the Assembly

The third stage of the assembly is posed as a material handling operation. The parts are completely put together; now the rigid product must then be transported to a specific location. We assume that the common load is beyond the load capacity of either manipulator, and that it must be transported by the two cooperating manipulators.

The third stage begins when the parts are completely put together by the cooperating manipulators, which can be detected by force sensors. Without stopping, they jointly transport the load to a specified location. The third stage ends when the cooperating manipulators place and release the load at the specified location, and return to the initial (parking) positions.

The overall system consisting of two manipulators holding the rigid product is modeled as a single closed kinematic chain mechanism. The configuration is shown in figure 1. For this configuration, six links can be seen; the 3rd link of each manipulator and the common load (rigid product) which is grasped are considered as one link, and in addition there are the two lower links of each manipulator, and the reference link [5]. There are six joints: three actuated joints for each manipulator. By equation (8):

$$\mathrm{DOF} = 3(6-1) - 2(6) = 3$$

Thus, the degrees of freedom of the total system are reduced by three when both manipulators grasp the rigid product. Thus, N=3 in equations (9-10) for the configuration under discussion. The three holonomic position, velocity and acceleration constraints have been obtained but are omitted due to space limitations.

The manipulators are dynamically coupled when they both grasp the rigid product. The forces and torques exerted by each manipulator on the load are not independent. Consider the free body diagram of the rigid common load shown in figure 3. The force and torque balance equations for this case are given in equations (11) - (13) and show the dependence.

$$ {}^1f_{3,4x} + {}^2f_{3,4x} = m_\ell \ddot{x}_{\ell\,\mathrm{c.m.}} \tag{11}$$

$$ {}^1f_{3,4y} + {}^2f_{3,4y} = m_\ell \ddot{y}_{\ell\,\mathrm{c.m.}} + m_\ell g_y \tag{12}$$

$$ {}^1n_{3,4} + {}^2n_{3,4} + r_{\ell\,ax}\,{}^1f_{3,4y} - r_{\ell\,ay}\,{}^1f_{3,4x} + r_{\ell\,ay}\,{}^1f_{3,4y} - r_{\ell\,by}\,{}^2f_{3,4x} = I_\ell\,\alpha_\ell \tag{13}$$

$$ x_{\ell\,\mathrm{c.m.}} = \ell_1\cos(\theta_1) + \ell_2\cos(\theta_1+\theta_2) + [\ell_3 + \frac{1}{2}(\ell_4+h_4)]\cos(\theta_1+\theta_2+\theta_3) \tag{14}$$

$$ y_{\ell\,\mathrm{c.m.}} = \ell_1\sin(\theta_1) + \ell_2\sin(\theta_1+\theta_2) + [\ell_3 + \frac{1}{2}(\ell_4+h_4)]\sin(\theta_1+\theta_2+\theta_3) \tag{15}$$

$$ r_{\ell\,ax} = -\frac{1}{2}(\ell_4+h_4)\cos(\theta_1+\theta_2+\theta_3)\ ;\ r_{\ell\,ay} = -\frac{1}{2}(\ell_4+h_4)\sin(\theta_1+\theta_2+\theta_3) \tag{16}$$

$$ r_{\ell\,bx} = \frac{1}{2}(\ell_4+h_4)\cos(\theta_1+\theta_2+\theta_3)\ ;\ r_{\ell\,by} = \frac{1}{2}(\ell_4+h_4)\sin(\theta_1+\theta_2+\theta_3) \tag{17}$$

$$ \alpha_\ell = (\ddot{\theta}_1 + \ddot{\theta}_2 + \ddot{\theta}_3) \tag{18}$$

where $\vec{g} = [0,-g_y]^T$ is the gravity vector; m_ℓ is the load mass, ${}^1f_{3,4x},{}^1f_{3,4y},{}^1n_{3,4})$ and $({}^2f_{3,4x},{}^2f_{3,4}y,{}^2n_{3,4})$ are the external forces and torque acting on arm #1 and arm #2, respectively. $(\theta_1,\theta_2,\theta_3)$ and (ℓ_1,ℓ_2,ℓ_3) are the arm #1 joint angles and link lengths, respectively. $(\ell_4 + h_4)$ is the length of the load.

This dependence enables us to again assume how the common load is distributed between the manipulators, i.e., we assume that the fractions of the total forces and torques exerted by the common load on each manipulator are specified. This assumption renders the equations obtained by free body analysis of the common load a determinate system. This can be demonstrated by considering the balance equations (11) - (13) obtained from figure 3. The three equations contain six unknowns: ${}^1f_{3,4x}$, ${}^1f_{3,4y}$, ${}^1n_{3,4}$, ${}^2f_{3,4x}$, ${}^2f_{3,4y}$, and ${}^2n_{3,4}$. Hence, the three equations (11) - (13) represent an indeterminate system.

Now assume that each manipulator assumes a specified fraction of the total load (forces and torques). Specifically, assume that the exerted forces and torques are proportional, as shown in equation (19).

$$ {}^1\vec{f}_{3,4} = k_1\,{}^2\vec{f}_{3,4}\ ;\ {}^1n_{3,4} = k_2\,{}^2n_{3,4} \tag{19}$$

which implies: ${}^1f_{3,4x} = k_1\,{}^2f_{3,4x}$; ${}^1f_{3,4y} = k_1\,{}^2f_{3,4y}$; where k_1 is the force distribution fraction and k_2 is the torque distribution fraction.

By substituting equation (19) into equations (11) - (13) we obtain:

$$ {}^2f_{3,4x} = \left[\frac{1}{1+k_1}\right] m_\ell \ddot{x}_{\ell\,\mathrm{c.m.}}\ ;\ {}^2f_{3,4y} = \left[\frac{1}{1+k_1}\right] [m_\ell \ddot{y}_{\ell\,\mathrm{c.m.}} + m_\ell g_y] \tag{20}$$

$$^{2}n_{3,4} = \left[\frac{1}{1+k_2}\right]\left[(r_{\ell\,by} + k_1 r_{\ell\,ay})\,^{2}f_{3,4x} - (r_{\ell\,bx} + k_1 r_{\ell\,ax})\,^{2}f_{3,4y} + I_\ell \alpha_\ell\right] \tag{21}$$

The system of equations (20) - (21) describes a determinate system because there are three equations in three unknowns ($^{2}f_{3,4x}, ^{2}f_{3,4y}, ^{2}n_{3,4}$). Hence, $^{1}f_{3,4x}$, $^{1}f_{3,4y}$, and $^{1}n_{3,4}$ can be determined by equations (19). The fractions apply for parallel transport of the common load.

With knowledge of the load parameters (mass, length, and inertia) and load distribution fractions, the exerted forces and torques can be determined and incorporated into the dynamical model of the closed kinematic chain mechanism. Thus, the mathematical model for the closed chain configuration of the third stage is obtained.

The load distribution assumption is reasonable, since for any control scheme using force servoing, the desired forces and torques exerted on the common load must be prespecified. Our approach provides insight into these desired forces and torques.

Upon releasing the load at the specified location, both manipulators operate independently in returning back to the initial (parking) positions. The assembly system again has six degrees of freedom.

IV. CONTROL CONSIDERATIONS

The degrees of freedom for the various configurations will determine how the two manipulator assembly system can be controlled. For the open chain configurations, the manipulators will be controlled independently. For the closed chain configurations, where the manipulators are coupled, a control scheme in which the overall system is considered as a single mechanism will be developed. Thus the overall control scheme must be able to detect the particular configuration of the two manipulator assembly system at any given instant of time and control it accordingly.

One way for the system controller to detect the current configuration is to equip each cooperating manipulator with a wrist force sensor. Consider the second stage where initially the loaded grippers are a certain distance apart and moving toward each other. The force sensors would detect the initial contacting of the parts, thus commencing closed chain control. Similarly, the force sensors would detect when the parts are completely put together, commencing control of the closed chain configuration of the third stage.

Adaptive control theory with self-tuning will be applied to the control of the two manipulator assembly system. Adaptive control has been demonstrated to successfully control the gross motion and constrained fine motion of a robotic manipulator [7,8].

IV.1. Control Considerations of the Open Chain Assembly

The planar manipulators will be independently controlled during the open chain configurations. The system controller will, however, ensure a coordinated motion of the two manipulators, e.g., in the first stage the manipulators should move from the parking positions to pick up and position their respective parts for subsequent assembly at the same time.

As a preliminary investigation, the planar manipulators are controlled in the joint coordinate system. Independent joint control is considered where the dynamics of each joint are modeled by an autoregressive model with external excitation (ARX (n,m)) given in equation (22).

$$\begin{aligned} y_i(k+1) &= a_{oi} + a_{1i}y_i(k) + a_{2i}y_i(k-1) + \cdots + a_{ni}y_i(k-n+1) \\ &\quad + b_{1i}u_i(k) + b_{2i}u_i(k-1) + \cdots + b_{mi}u_i(k-m+1) + e_i(k+1) \end{aligned} \tag{22}$$

where $y_i(.)$ is the angular position of joint i. $u_i(.)$ is the input to joint i. $e_i(.)$ is the modeling error, a zero mean, Gaussian white noise with variance ρ. $[a_{oi}, a_{1i}, \cdots, a_{ni}, b_{1i}, \cdots, b_{mi}]$ are the unknown parameters to be estimated. a_{oi} is a forcing term, which includes the effects of the gravitational forces.

The adaptive controller of the self tuning type is designed to minimize a conditional expectation of a performance functional given in equation (A.2) in the appendix. The minimizing controller for joint i is designed based on the ARX (m,n) model given in equation (22) with estimates substituted for the unknown parameter values. The parameters are estimated on-line using recursive least squares estimation [7]. The minimization of (A.2) with respect to the control input leads to the adaptive controller with self-tuning, given in equation (A.3).

Simulation studies have been performed for the motion of each 3 link planar manipulator carrying a held part during the first major stage of the assembly. Independent joint control was used; the dynamics of each joint were described by an ARX (2,1) model. A minimum variance position controller was designed for each joint ($w_2 = 0$ in (A.2)). The desired path for each joint was designed by assuming a trapezoidal velocity profile for the gripper, then obtaining the gripper's path in Cartesian coordinates. Given the desired gripper orientation at beginning and end of each segment along the desired path, the desired path in joint coordinates was obtained by inverse kinematics. Figures 4 and 5 show the motion of joint 2 along its desired path for arm #1 and arm #2 respectively.

IV.2. Control Considerations of the Closed Chain Assembly

The degrees of freedom for the closed chain configurations of the second and third assembly stages will determine how the two manipulator assembly system can be controlled. In the closed chain configuration of the second stage where relative motion is permitted between the parts, two degrees of freedom were lost due to two holonomic position constraints. Consequently, only 4 of the 6 actuated joints in the system can be controlled independently. In the closed chain configuration of the third stage where no relative motion is permitted between the parts, three degrees of freedom were lost due to the three holonomic position constraints. Consequently, only 3 of the 6 actuated joints in the system can be controlled independently.

Given the number of joints that can be controlled for each closed chain configuration, we can choose any combination of these joints. We can choose, for example, to control all of the joints of planar arm #1, and some (or none) of the joints of arm #2. Applying this approach to the closed chain configuration of the third stage, arm #2 can be chosen to play a completely passive role. The required joint positions, velocities and accelerations for arm #2 are determined by the holonomic constraints and the motion is caused by forces and torques from manipulator one transmitted through the rigid object (product) and reactions forces at the joints.

The joints that are not controlled are modeled as frictionless pin joints with no torques transmitted between the links which are connected by such joints. The actuator external inputs would be grounded in a physical implementation. The joint variables for the uncontrolled joints can, in fact, be eliminated from the dynamical model for the given closed chain configuration describing the two manipulator assembly system. This is possible by expressing these variables in terms of the variables of the controlled joints using the holonomic constraint equations (9-10). For the open chain configurations, all six actuated joints in the system can be controlled independently.

We are presently simulating our model for the assembly of two parts by two cooperating manipulators. Future work will involve the generalization of the approach to the three dimensional workspace using two six-joint manipulators.

V. REFERENCES

[1] Y.F. Zheng and F.R. Sias, "Two Robot Arms in Assembly," *Proceedings of the 1986 IEEE Robotics and Automation Conference,* San Francisco, CA, April 1986.

[2] J.L. Nevins and D.E. Whitney, "Research Issues for Automatic Assembly," *Information Control Problems in Manufacturing Technology,* IFAC Symposium, Tokyo, Japan.

[3] M. Brady, *Robot Motion,* pp. 13-18, MIT Press, 1982.

[4] K.R. Symon, *Mechanics,* Addison-Wesley, 1953.

[5] A.S. Hall, *Notes on Mechanism Analysis,* BALT Publishers, 1981.

[6] O.T. Greenwood, *Principles of Dynamics,* Prentice-Hall, 1965.

[7] A.J. Koivo and T.H. Guo, "Adaptive Linear Controller for Robotic Manipulator," *IEEE Trans. on Automatic Control,* Vol. AC-28, No. 2, February 1983.

[8] A.J. Koivo, "Force-Position-Velocity Control with Self Tuning for Robotic Manipulators," *Proceedings of the 1986 IEEE Robotics and Automation Conference,* San Francisco, CA, April 1986.

VI. APPENDIX

1. Newton-Euler model for arm #1 can be obtained in the form:

$$\begin{bmatrix} {}^1n_{0,1} \\ {}^1n_{1,2} \\ {}'n_{2,3} \end{bmatrix} = D \begin{bmatrix} \ddot\theta_1 \\ \ddot\theta_2 \\ \ddot\theta_3 \end{bmatrix} + E \begin{bmatrix} (\dot\theta_1)^2 \\ (\dot\theta_2)^2 \\ (\dot\theta_3)^2) \end{bmatrix} + F \begin{bmatrix} (\dot\theta_1\dot\theta_2) \\ (\dot\theta_1\dot\theta_3) \\ (\dot\theta_2\dot\theta_3) \end{bmatrix} + G\vec{g} + H \begin{bmatrix} {}^1f_{3,4x} \\ {}^1f_{3,4y} \\ {}^1n_{3,4} \end{bmatrix} + B \begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \\ \dot\theta_3 \end{bmatrix} \qquad (A.1)$$

where $n_{i-1,i}$, i=1,2,3 is the torque transmitted from link i-1 to link i; $(\theta_1,\theta_2,\theta_3)$, $(\dot\theta_1,\dot\theta_2,\dot\theta_3)$, $(\ddot\theta_1,\ddot\theta_2,\ddot\theta_3)$ are the joint positions, velocities, and accelerations; ${}^1f_{3,4x}$, ${}^1f_{3,4y}$, ${}^1n_{3,4}$ are the external forces and torques active on arm #1; (D,E,F,G,H,B) are the coefficient matrices for the inertia, centripetal, coriolis, gravity, reaction force, and viscous friction terms, respectively.

2. Performance Functional for Adaptive Controller

$$I[u_i(k)] = E\{w_1(y_i(k+1) - y_i^d(k+1))^2 + w_2u_i^2(k) \mid \psi(k)\} \qquad (A.2)$$

where $y_i(.)$ is the angular position of point i, $y_i^d(.)$ is the desired angular position of joint i, $u_i(.)$ is the input to joint i. w_1, w_2 are non-negative weighting factors, and $\psi(k)$ represents the past measurements, $\{y_i(k), y_i(k-1), \cdots\}$ and past inputs $\{u_i(k-1), u_i(k-2), \cdots\}$.

3. Adaptive Controller of the Self Tuning Type

$$u_i(k) = \frac{-w_1}{w_1(\hat b_{1i}(k))^2 + w_2} [\hat a_{oi}(k) + \hat a_{1i}(k)y_i(k) + \cdots + \hat a_{ni}(k)y_i(k-n+1)$$
$$+ \hat b_{2i}(k)u_i(k-1) + \cdots + \hat b_{mi}(k)u_i(k-m+1) - y_i^d(k+1)] \qquad (A.3)$$

where the caret denotes the estimated value of the parameter. The parameters are estimated by the least squares method.

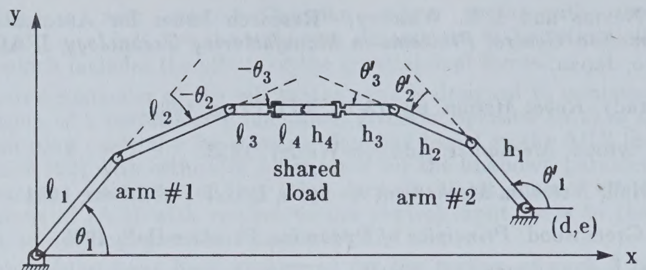

y
$-\theta_3$
θ'_3
θ'_2
$-\theta_2$
ℓ_2
ℓ_3
ℓ_4
h_4
h_3
h_2
h_1
shared load
θ'_1
ℓ_1
arm #1
arm #2
(d,e)
θ_1
x

Figure 1. Two Planar Manipulator Assembly System at the Third Stage

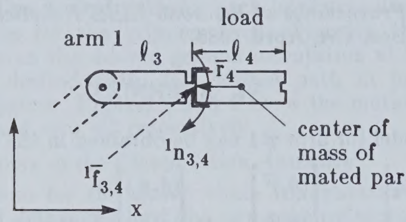

load
arm 1
ℓ_3
ℓ_4
$\bar{r}_4$
y
center of mass of mated part
${}^1n_{3,4}$
${}^1\bar{f}_{3,4}$
x

Figure 2. Freebody Diagram for Mated Part Held by Arm #1

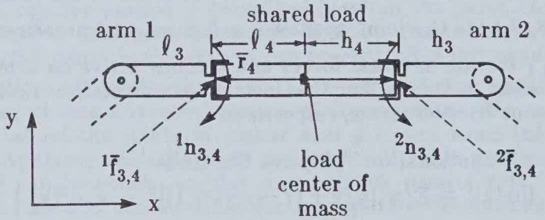

shared load
arm 1
ℓ_3
ℓ_4
h_4
$\bar{r}_4$
h_3
arm 2
y
${}^1n_{3,4}$
${}^2n_{3,4}$
${}^1\bar{f}_{3,4}$
${}^2\bar{f}_{3,4}$
load center of mass
x

Figure 3. Freebody Diagram for Rigid Shared Load

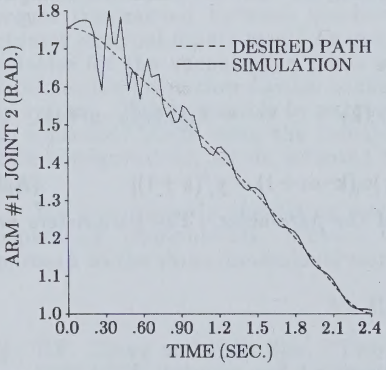


Figure 4. Trajectory of Joint 2, arm #1

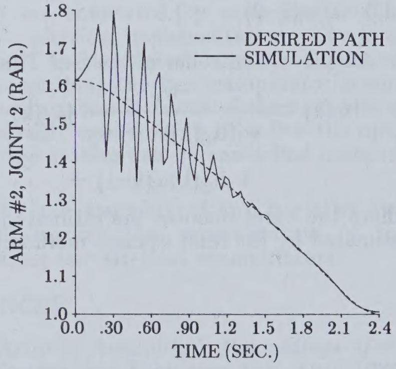


Figure 5. Trajectory of Joint 2, arm #2

DESIGN OF PI-REGULATORS FOR MANIPULATORS VIA SINGULAR PERTURBATION

FARSHAD KHORRAMI and ÜMIT ÖZGÜNER
Department of Electrical Engineering
The Ohio State University
2015 Neil Avenue
Columbus, Ohio 43210

ABSTRACT

Asymptotic tracking of a constant set point for an n-link manipulator via PI-controller and gravity cancellation is considered. Thereafter, a small parameter is introduced in the dynamics through the assumption that one link is *much heavier* than the other links and a singular perturbation approach is taken to design a composite control law.

1. INTRODUCTION

Development of control strategies for robot manipulators is considered in this paper. Linearization via non-linear feedback known as *inverse dynamics* or *computed torque* has been used extensively in the literature [1,2,3]. A number of drawbacks exist in practice due to the fact that the global stability of the method depends upon having exact dynamics of the system, accuracy, and speed in computation and ideal torque generators. Furthermore, degradation of response occurs due to noise in the system; therefore, the strategies such as adaptive [4]-[5] and/or variable structure control [6] have been suggested to compensate for the above problems. In the present paper simple control strategies such as gravity cancellation and PI control for asymptotic tracking of a set-point are introduced.

Designing control laws for systems with large dimension and non-linearities is cumbersome if not impossible. Furthermore, due to need for on-line implementation of control algorithms, it is of interest to reduce the system order and/or to simplify the system dynamics in some systematic manner: singular perturbation [7,8,9] may be a promising method to attack this problem. Basically, two reduced order subsystems, slow and fast, are derived through perturbation methods and a composite control is formed. The composite control is composed of the control for the slow subsystem and the control for the fast subsystem which is to stabilize the fast subsystem along its equilibrium trajectory created by the controlled slow subsystem. In the present paper, for brevity, this method is applied to a five-link manipulator (refer to Fig. 1.). However, the algorithm can be extended easily to an n-link manipulator in the same fashion.

2. SINGULAR PERTURBATION

The following discussion is based upon the assumption of the mass and inertia of one link being much larger than the other links. In the case of the five-link robot in question, the third link is assumed to be the aforementioned link. With the above assumption, a small parameter ϵ can be imbedded in the dynamics where ϵ is the ratio of second heaviest link to the third one.

The dynamics of a robot with open-chain kinematics is given as:

$$J(\theta)\ddot{\theta} + J_c(\theta)\dot{\theta}^2 = G(\theta) + BU \tag{1}$$

where J is the inertia matrix, G is the gravity vector, and $J_c(\theta)\dot{\theta}^2$ represents the Coriolis and centripetal forces.

We introduce the small parameter ϵ into the dynamics knowing that the third link has much larger mass and inertia as follows:

$$\begin{aligned} m_3 &= M \\ m_1 &= m_2 = m_4 = m_5 = M\epsilon \\ I_3 &= I \\ I_1 &= I_2 = I_4 = I_5 = I\epsilon \end{aligned} \tag{2}$$

where $\epsilon << 1$.

Substituting (2) in (1), one obtains

$$J(\theta,\epsilon)\ddot{\theta} + J_c(\theta,\epsilon)\dot{\theta}^2 = G(\theta,\epsilon) + BU \tag{3}$$

The term multiplying the highest derivative, $J(\theta,\epsilon)$, has rank deficiency at $\epsilon = 0$; therefore, the differential equation given by (3) is singularly perturbed.

Remark: The last two rows vanish in $J(\theta,\epsilon)$ for $\epsilon = 0$; hence, the last two links correspond to the fast subsystem. In general, if the mth link is the heaviest one, then the slow subsystem corresponds to the 1st through mth link and the rest of the links constitute the fast part.

To derive the slow and fast subsystems, partition the matrices in (1) as:

$$J(\theta) = \begin{bmatrix} J_{11\ \ 3\times3} & \vdots & J_{12} \\ \cdots\cdots\cdots & \vdots & \cdots\cdots\cdots \\ J_{21} & \vdots & J_{22\ \ 2\times2} \end{bmatrix} ; \quad J_c(\theta) = \begin{bmatrix} J_{c11\ \ 3\times3} & \vdots & J_{c12} \\ \cdots\cdots\cdots & \vdots & \cdots\cdots\cdots \\ J_{c21} & \vdots & J_{c22\ \ 2\times2} \end{bmatrix} \tag{4a}$$

$$G(\theta) = \begin{bmatrix} G_{1\ \ 3\times1} \\ \cdots\cdots\cdots \\ G_{2\ \ 2\times1} \end{bmatrix} ; \quad B = \begin{bmatrix} B_{11\ \ 3\times3} & \vdots & B_{12} \\ \cdots\cdots\cdots & \vdots & \cdots\cdots\cdots \\ B_{21} & \vdots & B_{22\ \ 2\times2} \end{bmatrix} \tag{4b}$$

Define

$$\begin{aligned} Z_1 &= [\theta_1\ \theta_2\ \theta_3]' \ ; \quad Z_2 = [\theta_4\ \theta_5]' \\ U_s &= [U_1\ U_2\ U_3]' \ ; \quad U_f = [U_4\ U_5]' \end{aligned} \tag{5}$$

where (') denotes transpose.

Using the above partition and (2), multiply out the terms in (3) and obtain

$$\begin{cases} J_1\ddot{Z}_1 + \epsilon J_{12}\ddot{Z}_2 + J_{c_1}\dot{Z}_1^2 + \epsilon J_{c_{12}}\dot{Z}_2^2 = G_{11} + B_{11}u_s + B_{12}u_f \\ \epsilon J_{21}\ddot{Z}_1 + \epsilon J_{22}\ddot{Z}_2 + \epsilon J_{c_{21}}\dot{Z}_1^2 + \epsilon J_{c_{22}}\dot{Z}_2^2 = \epsilon G_2 + B_{22}u_f \end{cases} \tag{6}$$

Remark: $B_{21} = 0$ due to special structure of B. Furthermore, J_1, J_{c_1}, and G_{11} are $J_{11}, J_{c_{11}}$, and G_1 evaluated at (2).

To transform the above singularly perturbed system to a regularly perturbed one and obtaining the fast subsystem, define the stretched time scale τ as

$$\tau = \frac{t}{\sqrt{\epsilon}} \tag{7}$$

Therefore,

$$\begin{cases} \frac{d}{dt} = \frac{1}{\sqrt{\epsilon}}\frac{d}{d\tau} \\ \frac{d^2}{dt^2} = \frac{1}{\epsilon}\frac{d^2}{d\tau^2} \end{cases} \tag{8}$$

Using (8), rewrite (6) in the following manner:

$$\begin{cases} \frac{1}{\epsilon}J_1\ddot{Z}_1 + J_{12}\ddot{Z}_2 + \frac{1}{\epsilon}J_{c_1}\dot{Z}_1^2 + J_{c_{12}}\dot{Z}_2^2 = G_{11} + B_{11}u_s + B_{12}u_f \\ J_{21}\ddot{Z}_1 + J_{22}\ddot{Z}_2 + J_{c_{21}}\dot{Z}_1^2 + J_{c_{22}}\dot{Z}_2^2 = \epsilon G_2 + B_{22}u_f \end{cases} \tag{9}$$

where $(\cdot)$ denotes differentiation with respect to τ. To obtain the slow and the fast subsystems, expand the terms in (6) and (9) in powers of ϵ; thereafter, collect the leading order terms. Therefore,

Slow Subsystem

$$J_1(\bar{Z}_1,0)\ddot{\bar{Z}}_1 + J_{c_1}(\bar{Z}_1,0)\dot{\bar{Z}}_1^2 = G_{11}(\bar{Z}_1,0) + B_{11}U_s \tag{10}$$

Fast Subsystem

$$\begin{cases} J_1(\bar{Z}_1,0)\ddot{\bar{Z}}_1 + J_{c_1}(\bar{Z}_1,0)\dot{\bar{Z}}_1^2 = 0 \\ J_{21}(\bar{Z}_1,\tilde{Z}_2,0)\ddot{\bar{Z}}_1 + J_{22}(\tilde{Z}_2,0)\ddot{\tilde{Z}}_2 + J_{c_{21}}(\bar{Z}_1,\tilde{Z}_2,0)\dot{\bar{Z}}_1^2 \\ \qquad + J_{c_{22}}(\bar{Z}_1,\tilde{Z}_2,0)\dot{\tilde{Z}}_2^2 = B_{22}U_f \end{cases} \tag{11}$$

$$\Longrightarrow \quad J_{22}(\tilde{Z}_2,0)\ddot{\tilde{Z}}_2 + J_{c_{22}}(\bar{Z}_1,\tilde{Z}_2,0)\dot{\tilde{Z}}_2^2 = B_{22}U_f \tag{12}$$

To obtain equation (12), it should be noted that the derivative of the slow variable with respect to the fast time scale vanishes. Thereafter, Examining (10) and (12), one can conclude that the slow subsystem corresponds to the 1st through the heaviest link and the fast subsystem corresponds to the remaining links parameterized by the slow variables.

3. GRAVITY CANCELLATION AND PI-REGULATOR

Applying singular perturbation technique to the interconnected rigid bodies, the gravity terms in the equation of motion vanish for the fast subsystem. The resulting dynamic equation is

$$J(\theta)\ddot{\theta} + J_c(\theta)\dot{\theta}^2 = BU \tag{13}$$

The above system (13) is stabilizable by linear state feedback; i.e.,

$$u = B^{-1}\{K_1(\theta - \theta_{ref}) + K_2\dot{\theta}\} \tag{14}$$

where θ_{ref} is a constant set point. The proof is based upon choosing the following Lyapunov function

$$V(t) = \frac{1}{2}\left[(\eta_1(t) - \theta_{ref})' Q_1(\eta_1(t) - \theta_{ref}) + \eta_2'(t) J(\eta_1)\eta_2(t)\right]$$

where

$$\begin{aligned} \eta_1 &= \theta \\ \eta_2 &= \dot{\theta} \end{aligned} \tag{15}$$

Remark: The full dynamics of an n-link robot (including the gravity terms) can also be stabilized by linear state feedback. This can be shown by adding a potential term to the Lyapunov function.

Examining the equilibrium point in the error state space (η_1, η_2), it can be observed that a steady state error exists in the joint variable due to gravity unless the components of θ_{ref} are integer multiples of π (i.e., $n\pi$ for $n = \pm 1, \pm 2, \ldots$). This problem can be avoided by either gravity cancellation or PI regulation.

One can cancel only the gravity terms rather than using the inverse plant as a feedforward compensator. Furthermore a linear state feedback can be added for stabilization purposes; i.e.,

$$U(t) = B^{-1}[-G(\eta_1) + K_1(\eta_1 - \theta_{ref}) + K_2\eta_2] \tag{16}$$

Using the same Lyapunov function as before, it can be shown that the plant is stabilized and furthermore, the origin is the equilibrium point in the error state space set up by this control, i.e.,

$$\theta - \theta_{ref} \longrightarrow 0 \quad \text{as} \quad t \longrightarrow \infty$$

However, one still can argue that parameter variations in the gravity vector will not permit exact cancellation and may cause some error. The approach advocated is the use of a dynamic feedback servocompensator, i.e., Proportional-Integral feedback. This feedback strategy insures proper biasing torque despite uncertainties in the parameters of the link and the mass of the load.

The PI-controller has the form

$$u = B^{-1}\{K_1(\eta_1 - \theta_{ref}) + K_2\eta_2 + K_3\eta_3\} \tag{17a}$$

where

$$\dot{\eta}_3 = \eta_1 - \theta_{ref} \tag{17b}$$

Using the above feedback structure, the closed-loop system dynamics are

$$\begin{cases} \dot{\eta}_1 = \eta_2 \\ \dot{\eta}_2 = J^{-1}\{-J_c\eta_2^2 + G + K_1\eta_1 + K_2\eta_2 + K_3\eta_3\} \\ \dot{\eta}_3 = \eta_1 - \theta_{ref} \end{cases} \tag{18}$$

Choose the following Lyapunov function:

$$\begin{aligned} V(t) &= \frac{1}{2}\left(\eta_1'Q\eta_1 + \eta_2'J(\xi_1)\eta_2 - \theta_{ref}'Q\theta_{ref}\right) \\ &+ [G(\theta_{ref}) - Q\theta_{ref}]'(\eta_1 - \theta_{ref}) - \int_a^t \eta_2'G(\eta_1)\, ds \end{aligned} \tag{19}$$

The following choice of feedback gains insures global asymptotic stability of the system:

$$\begin{aligned} K_1 &= -Q < 0 && (20) \\ K_2 &< 0 && (21) \\ K_3 &= \gamma I && (22) \end{aligned}$$

where $\gamma < \frac{\eta_2'G(\theta_{ref})}{\eta_2'\eta_3}$ and Q is a diagonal matrix with diagonal elements q_{ii}.

4. COMPOSITE CONTROL

As manifested by the two reduced order subsystems, the feedback control law design can be split into two separate designs for the reduced order subsystems, i.e., slow and fast. The slow subsystem corresponds to the dynamics of a three-link manipulator. One could use any of the control strategies such as model reference adaptive control, inverse plant, and PI-controller depending on the objectives.

The fast subsystem can be stabilized with the slow variable as a parameter by nonlinear feedback cancellation or any other strategies suggested before. Furthermore, the composite control can be established by adding the controls for the slow and fast subsystems, namely

$$U(t) = U_s(\bar{Z}_1) + U_f(\bar{Z}_1, \tilde{Z}_2)$$

such that $U_f(\bar{Z}_1, 0) = 0$; i.e., the control U_f is inactive along the solution of the slow subsystem.

5. CONCLUSION

Control strategies for asymptotic tracking of a constant set point for an n-link manipulator are discussed. It is shown that using either the gravity cancellation technique or the PI-regulator, one can get rid of steady state errors on the joint variables caused by state feedback only. Although the proof guarantees asymptotic tracking of a constant set point, one may hope that the system still performs *satisfactorily* if the constant reference signal is replaced by a slowly time-varying signal (i.e., *small* derivatives).

Furthermore, assuming that the manipulator has a link that is *much heavier* than the other links, singular perturbation technique was applied to reduce the system order and dynamics. Ultimately a composite control was designed to regulate the system. Simulation studies for a 5-link manipulator are being performed. Further work on obtaining estimates of the perturbation constant, ϵ, is of interest to find out for what values of ϵ the above technique yields a *satisfactory* result.

The present approach is useful for manipulators where one of the intermediate links is heavier than the rest and multi-link walking models for animals where the *body* has greater mass than the *legs*. The application of the above analysis to a model of a jumping quadruped will be presented elsewhere.

Because of lack of space some proofs are not included and may be obtained from the authors.

6. ACKNOWLEDGMENT

This material is based upon work supported by the Department of Defense under contract DAAE07-84-K-R001. The authors are greatful to Professor Yuji Kodama and Dr. Helmut Buchner, The Ohio State University, for their valuable discussions.

7. REFERENCES

[1] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, "On-line computational scheme for mechanical manipulators," *Trans. ASME J. Dyn. Syst. Measurement Contr.*, pp. 69–76, June 1980.

[2] M. Brady et. al. (editor), *Robot motion: planning and control*, MIT Press, 1982, (chapter 3: Feedback control).

[3] R. P. C. Paul, *Robot manipulators: mathematics, programming, control*, MIT Press, Cambridge, MA, 1981.

[4] F. Al-Abbass, *Decentralized model reference adaptive systems with variable structure control*, Ph.D. Dissertation, The Ohio State University, Summer 1986.

[5] S. Dubowsky, D. T. Desforges, "The application of model-referenced adaptive control to robotic manipulator," *Journal of Dynamic Systems, Measurement, and Control*, vol. 101, pp. 193-200, Sep. 1979.

[6] R. G. Morgan, Ü. Özgüner, "A decentralized variable structure control algorithm for robotic manipulators," *IEEE Journal of Robotics and Automation*, vol. RA-1, no. 1, pp. 57–65, March 1985.

[7] P. V. Kokotovic, "Application of singular perturbation techniques to control problems," *Siam Review*, vol. 26, no. 4, pp. 501–550, October 1984.

[8] V. R. Saksena, J. O'Reilly, and P. V. Kokotovic, "Singular perturbations and time-scale methods in control theory: survey 1976-1983," *Automatica,* vol. 20, no. 3, pp. 273–293, 1984.

[9] Ali H. Nayfeh, *Perturbation Methods*, Wiley Interscience, New York, 1973.

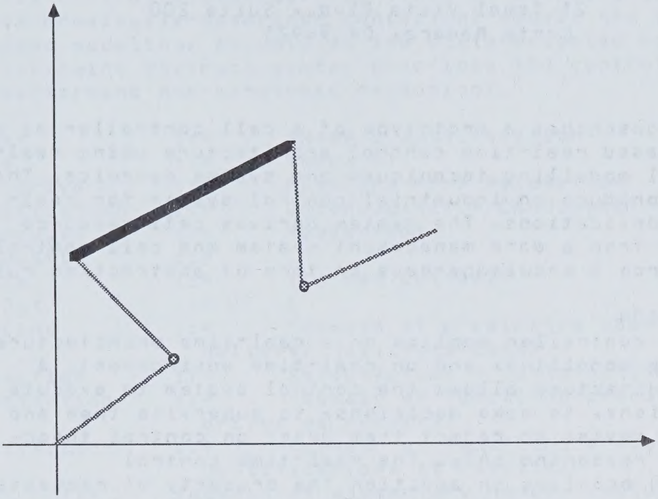


Fig. 1: 5-link manipulator

A REAL-TIME KNOWLEDGE-BASED CELL CONTROLLER

I. Mina

MAXITRON Corp.
21 Tamal Vista Blvd., Suite 200
Corte Madera, CA 94925

ABSTRACT

This paper describes a prototype of a cell controller as a knowledge-based real-time control architecture using real-time control modelling techniques and system dynamics. The aim was to produce an industrial control system for real-time cell applications. The system derives cell resource information from a data management system and cell control knowledge from a knowledge-base in form of abstraction rules.

1 INTRODUCTION

A real-time controller applies on a real-time architecture, on real-time modelling, and on real-time environment. A control architecture allows the control system to execute control actions, to make decisions, to supervise them and to maintain or revise or reject them based on control theory and control reasoning only. The real-time control architecture provides in addition the property of requesting or responding to control activities immediately. Immediately means here "without culpable delay" [1].

2 THE CELL CONTROLLER

The goal of the research was to design and implement a knowledge-based real-time cell controller which can describe the cell environment and the tasks to be controlled, build a knowledge base for cell control build reasoning components and the modeling concept for cell control, and build an architecture for cell control.

The knowledge representation and reasoning in this cell controller is based on XPS85 [5], a real-time language framework for qualitative, spatial, temporal, causal and non-monotonic reasoning on technical systems. Both exercising control and reasoning about justifications can be formulated in terms of meta-rules which test on their left-hand sides for properties of e.g.

(control:) -- previous derivation history
-- applicable rule-belief pairs
-- the current state of beliefs, i.e., the sets IN and OUT

(inference on justifications:) -- justifications.

The actions of such meta-rules then consist in

(control): suggesting which applicable rule-belief pair is to be executed.

(inference on justifications): inferring beliefs with justifications which are deduced from justifications in other beliefs [2]. This may often appear as if actually justifications in beliefs are replaced with upgraded inferred justifications.

In this control prototype, two types of modeling apply based on the previously-described conceptual model: the event-oriented modeling, as well as the state-oriented modeling. The following abstract syntax describes the control system for performing non-monotonic reasoning:.

```
Belief            ::=     <Name L Justification>
Drive             ::=     (Name R)
Meta-Drive        ::=     MetaCond ---> MetaAction
Justification     ::=     UncondJustif + CondJustif
UncondJust        ::=     Reason-set
CondJustif        ::=     L
Reason            ::=     (In-set Out-set)
In, Out           ::=     L
MetaCond          ::=     {domain of predicates about drives,
                       beliefs, justifications}
MetaAction        ::=     SelectAction + JustifAction
SelectAction      ::=     {domain of functions producing
                       drives as results}
JustifAction      ::=     {domain of functions producing
                       justifications as results}
Name    ::=     {domain of elementary names, including the
             empty string}.
```

The knowledge-base of this real-time cell controller consists of abstraction rules on various aspects considering the objects to be controlled, the control operations to be activated, the control processes to be executed, the tasks to be performed. In the individual phases the following steps are to be taken: Problem Structure, Analysis of Histories, Development of Principle Solutions, Evaluation of Principle Solutions, Concretization, Development of Conceptional Variants, Evaluation and Iteration.

Computer assistance of this control architecture is achieved by the following set of software tools: XPS85-EDITOR, ANALYZER, ADVISOR, DETAIL-PLANNER, INTERPRETER, EVALUATOR. The real-time control architecture associates the above described control phases with these tools.

3 SUMMARY

The objective of the above represented knowledge-based manufacturing control system is to contribute towards mechanizing control with respect to the planning and communication activities.

4 REFERENCES

1 Mina, I.: Knowledge-based Cell Control. KBSDPublications, 140 University Ave., Palo Alto, CA 94301, 1986.

2 Raulefs, P.: A real-time architecture for knowledge-based control. KBS-publications, Berlin, 1985.

3 XPS85: A language for qualitative, temporal, causal, non-monotonic reasoning and modelling. KBS-Publications, Berlin, 1985.

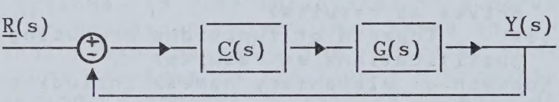
COMPUTER-AIDED DESIGN OF MULTIVARIABLE CONTROL SYSTEMS WITH APPLICATIONS TO AIRCRAFT FLYING QUALITIES*

L.R. PUJARA
Dept. of Electrical Systems Engineering,
Wright State University,
Dayton, OH 45435

ABSTRACT

In this paper, we propose a computer-aided technique of multivariable control systems design by matching the frequency-responses of the compensated closed loop transfer function matrix and a desired transfer function matrix. It is a generalization of the SISO technique by the author [1]. A similar idea of using a desired transfer function matrix for multivariable control system design has been used by Lehtomaki, et.al. [2]. The proposed work is in its preliminary stage but looks promising.

Consider the following block diagram of a typical multivariable feedback control system of order n:



Suppose that $\underline{D}(s)$ is an n x n desired closed loop transfer function matrix which has been synthesized to satisfy a given set of specifications. Define a weighted mean square error between the frequency responses of $\underline{D}(s)$ and $(I + \underline{G}(s)\underline{C}(s))^{-1}\underline{G}(s)\underline{C}(s)$ as follows:

$$E = \int_{\omega_1}^{\omega_2} \left[\sum_{i,k=1}^{n} \left| \lambda_{ik}(j\omega) - \sum_{r=1}^{n} \mu_{ir}(j\omega)\, d_{rk}(j\omega) \right|^2 \right] d\omega \qquad (1)$$

where

$$\lambda_{ik}(s) = \left[\sum_{r=1}^{n} g_{ir}(s) c_{rk}(s) \right] \cdot d(s), \quad \mu_{ik}(s) = k(s)g(s)\delta_{ik} + \lambda_{ik}(s) \qquad (2)$$

$$\underline{G}(s) = \frac{\left[g_{ik}(s)\right]}{g(s)}, \qquad \underline{D}(s) = \frac{\left[d_{ik}(s)\right]}{d(s)}, \quad \underline{C}(s) = \frac{\left[c_{ik}(s)\right]}{k(s)} \qquad (3)$$

$$c_{ik}(s) = \sum_{r=0}^{m(i,k)} a_r^{(i,k)} s^r, \quad k(s) = 1 + \sum_{r=1}^{p} b_r s^r, \quad m(i,k) \leq p, \quad 1 \leq i, k \leq n \tag{4}$$

δ_{ik} is the usual Kronecker delta and $[\omega_1, \omega_2]$ is the frequency interval of interest. The unknown controller-parameters are determined by minimizing E. The significance of defining E in this fashion is that the minimization of E leads to linear algebraic equations involving the controller parameters.

In applying the technique, the tricky problem is to synthesize a desired transfer function matrix. Recently*, we constructed the following desired transfer function matrix for the aircraft A-7A at M =0.6, h = 15,000 ft in its lateral directional mode and satisfying the flying qualitites criteria [3]:

$$\begin{bmatrix} \phi \\ \\ \beta \end{bmatrix} = \begin{bmatrix} \dfrac{4.2699(s^2+1.6s+16)}{s(s+1)(s^2+1.6s+16)} & \dfrac{6.8318}{s(s+1)(s^2+1.6s+16)} \\ & \\ 0 & \dfrac{6.0681s(s+1)}{s(s+1)(s^2+1.6s+16)} \end{bmatrix} \begin{bmatrix} \delta_a \\ \\ \delta_r \end{bmatrix} \tag{5}$$

where ϕ is the roll langle, β is the sideslip angle, δ_a is the aileron deflection and δ_r is the rudder deflection.

REFERENCES

1. L.R. Pujara, "A Frequency-Domain Method of Designing a Controller II," The IEEE Naecon Proceedings, 1986, pp. 450-454.

2. Norman Lehtomaki, G. Stein and Joseph Walls, Jr., "Multivariable Prefilter Design for Command Shaping," AIAA Conference, 1984, pp. 38-43.

3. D.J. Moorhouse and R. J. Woodcock, "Background Information and User Guide for MIL-F-8785C, Military Specifications Flying Qualities of Piloter Airplanes," AFWAL, WPAFB, Ohio, TR-3109, July, 1982.

*Part of this work was done while the author was a visiting scientist in Summer, 1986 at the Flight Dynamics Lab, Wright Patterson Air Force Base, Ohio under the AFOSR Summer Faculty Research Program.

VELOCITY FEEDBACK CONTROL AND THE DYNAMIC PROPERTIES OF ROBOT MANIPULATORS

FATHI M. A. SALAM

Department of Electrical Engineering and Systems Science
Michigan State University
East Lansing, Michigan 48824

SUMMARY

The recently proposed control schemes that rely on linearizing-by-cancellation nonlinear controllers suffer from the ever-present mismatch between the model and the actual dynamics of a robot manipulator. Two factors contribute to this mismatch: (1) errors in modeling the actual dynamics, and (2) errors in computation when implementing the nonlinear controller [1,2,3].

We observe that these schemes are applicable to all manipulators irrespective of their design parameters (e.g. the length and the mass of the links, etc.). In this sense, these schemes divorce the control problem from the design problem.

For position and velocity control, we show that employing linear PD [1,2] controllers is more advantageous than the nonlinear cancelling control approach. Moreover, we show that employing a simpler controller form (namely, D controllers) performs comparatively well [4].

As a consequence, the second linear controller underlines the connection between the design stage and the control stage of robot manipulators. A design that improves the dynamics would minimize the control effort and limit it to mere stabilization. Hence the resulting dynamic paths of the manipulator are shaped by the inherent design parameters. In that sense they are viewed as "natural" paths of the manipulator.

The potential function holds the key to the design of the manipulator. The selection of the links parameters such as length and mass contributes to the structure of the potential function. Also because of practical and physical limitations, one is only interested in the joint angle traversing within 2π-interval centered about the desired final position.

This work entails the presentation of ideas and observations that connect the design of manipulators and their control. It addresses the point-to-point control using velocity feedback and proves the convergence of a well characterized set of initial velocities and positions to a desired

final position of the manipulator's joints [4]. However, one cannot ensure that all initial velocities and positions will converge [5]. It also addresses design by focusing on parameter selection in order to improve the dynamic properties of manipulators. In the focus on design, the function of the controller becomes merely that of stabiliztion.

REFERENCE

[1] M. Takegaki and S. Arimoto, "A New Feebback Method for Dynamic Control of Manipulators," Journal of Dynamic Systems, Measurement, and Control, Vol. 102, June 1981, pp. 119- 125.

[2] M. Takegaki and S. Arimoto, "Stability and Robustness of PID Feedback Control for Robot Manipulators of Sensory Capability," Robotics Research, M. Brady and R. Paul (editors), MIT Press, 1984, pp. 783-799.

[3] D. Koditschek, "Natural Motion for Robot Arms," Proceedings of the 23rd IEEE Conference on Decision and Control, Las Vegas, Nevada, Dec. 12-14, 1984, pp. 733-735.

[4] F.M.A. Salam, "On The Design and Control of Robot Manipulators," 1986 International Conference on Systems, Man and Cybernetics, Atlanta, Georgia, October 14-17,1986.

[5] F.M.A. Salam, "Feedback Stabilization of The Nonlinear Pendulum Under Uncertainty: A Robustness Issue," Systems and Control Letters, Vol. 7, no. 3, pp. 199- 206.

MEMORY REDUCTIONS FOR ROM BASED LOGARITHMIC CONVERSION

S.C. LEE AND T. MIGIMATSU
University of Oklahoma
Norman, OK 73019

ABSTRACT

As a part of the next generation computers, the logarithmic number system is an attractive feature since it is capable of high speed arithmetic operations with accuracy [1]. Despite this capability, development of this system is slow due to the complexity of the logarithms.

One major issue for developing the logarithmic number system is the conversion of the numbers. A table look-up method using Read Only Memory (ROM) is one approach for the conversion [2]. This method generally increases the conversion speed with accuracy.

Tracing back the history of the logarithmic number conversion methods, probably the earliest and simplest method without using ROM is the Mitchell's [3] approximation method. Since Mitchell's [3] development in 1962, some improvements have been achieved [4] - [6]. Although a ROM conversion method has been suggested by Kingsbury and Rayner [2], it has not been explored fully while the ROM arithmetic methods have been demonstrated [7] - [11].

This paper presents useful formulae for determining the length of a logarithm in terms of desired accuracy. This formulation regulates the word length of a ROM based logarithmic conversion table. In addition to this formulation, two efficient table reduction methods are proposed: Upper-Cut table reduction and Mirror-Image table reduction. Upper-Cut table reduction is capable of the logarithmic addition/subtraction. Mirror-Image table reduction is adequate for the base-two logarithm. Due to successful reduction size, simplicity of manipulation and a high degree of speed with accuracy, these reduction methods are sufficient to accommodate the conversion table for the logarithmic number system.

This paper further proposes a BCD (base-10) logarithm as a new implementation of the logarithms applied to the Upper-Cut reduction method.

REFERENCES

1. Tamio, Krokawa, James A. Payne, and Samuel C. Lee, "Error Analysis of Recursive Digital Filters Implemented With Logarithmic Number Systems", IEEE Trans. on Acoustics, Speech, and Signal Processing, Vol. ASSP-28, No. 6, Dec. 1980.

2. N.G. Kingsbury and P.J.W. Rayner, "Digital Filtering Using Logarithmic Arithmetic", Electron. Lett., Vol. 7, pp. 56-58, Feb. 1971.

3. John N. Mitchell, "Computer Multiplication and Division Using Binary Logarithms", IRE Trans. Electron. Comput., Vol. EC-11, pp. 512-517, Aug. 1962.

4. M. Combet, H. Van Zonneveld, and L. Verbeek, "Computation of the Base Two Logarithm of Binary Numbers", IEEE Trans. Comput., Vol. EC-14, pp. 863-867, Dec. 1965.

5. E.L. Hall, D.D. Lynch, and S.J. Dwyer, "Generation of Products and Quotients Using Approximate Binary Logarithms for Digital Filtering Applications", IEEE Trans. on Comput., Vol. C-19, No. 2, pp. 97-105, Feb. 1970.

6. D. Marino, "Algorithms for the Approximate Evaluation in Hardware of Binary Logarithms and Elementary Functions", IEEE Trans. on Comput., Dec. 1972.

7. J.C. Majithia and D. Levan, "A Note on Base-2 Logarithm Computations", Proc. IEEE (Lett.), Vol. 61, pp. 1519-1520, Oct. 1973.

8. Thomas A. Brubaker and John C. Becker, "Multiplication Using Logarithms Implemented with Read-Only Memory", IEEE Trans. Comput., Vol. C-24, No. 8, pp. 761-765, Aug. 1975.

9. Albert D. Edgar and Samuel C. Lee, "FOCUS Microcomputer Number System", Comm. of the ACM, Vol. 22, N-3, pp. 166-177, Mar. 1979.

10. Earl E. Swartzlander, Jr. and Alexopoulos G. Aristides, "The Sign/Logarithm Number System", IEEE Trans. Comput., Vol. C-24, pp. 1238-1242, Dec. 1980.

11. Earl E. Swartzlander, Jr., D.V. Satish Chandra, H. Troy Nagle, Jr. and Scott A. Starks, "Sign/Logarithm Arithmetic for FFT Implementation", IEEE Trans. Comput., Vol. C-32, pp. 526-534, June 1983.

A MINIMUM SPANNING TREE BASED ALGORITHM FOR DELAY-BOUNDED ARRAY RECONFIGURATION (*)

DONATELLA SCIUTO

Department of Electrical and Computer Engineering,
University of Colorado (USA)

(*) This work was supported by a grant from AT&T and NATO

INTRODUCTION

The paper deals with the implementation of fault tolerant architectures using WSI technology. One of the approaches to design with WSI technology is the repeated use of identical processing elements organized into arrays or tree connections. Local interconnections between processing elements are essential to lower propagation delays and to reduce interconnection complexity. A fault tolerant procedure allowing reconfiguration in presence of faults of such kind of devices should minimize interconnection delays.

The proposed algorithm represents a unified approach to two different types of WSI array reconfiguration:

1) static reconfiguration, usually performed at production time on large wafer scale devices and mainly actuated by irreversible actions (e.g. laser cutting or microfuses blowing);

2) externally driven dynamic reconfiguration, usually executed the host computer during the operational life of the device when random faults are more likely to occur. In this case some additional reconfiguration control logic, such as programmable switches, is required to bypass faulty elements.

THE RECONFIGURATION ALGORITHM

Aim of the proposed algorithm is to achieve fast reconfiguration of an array minimizing interconnection delays between reconfigured elements. The bidimensional array is constituted by an operational array and a number of spare rows and columns. The reconfiguration algorithm allows to map an operational array on the overall array in presence of faults.

Let's consider the array as an undirected connected graph G(V,E), where V is the set of vertices and E the set of edges. Each processing element, identified by its position in the array (i,j) is a vertex of the graph, while each interconnection corresponds to an edge. The algorithm is based on the parallel construction of minimum spanning trees in order to minimize communication delays between fault-free elements.

The algorithm can be subdivided into three logical steps:

1) Partition of the operational array into subarrays. Partition consists of subdividing the operational array into a number of subarrays S_a $(1 \leq a \leq k)$, which are defined such that the intersection between adjacent subarrays is not void. This type of partition allows to avoid the definition of an interconnection procedure between the subarrays independently reconfigured.

2) Subarray compensation. Prior to reconfiguration, mapping of indices is necessary in order to allow convergence of the interconnection's reconfiguration, that is to make sure that the array is reconstructed in its original shape. A heuristic algorithm is executed on the entire array in order to achieve a balance on the number of fault-free elements in each subarray.

3) Define interconnections. For each subarray the following steps are performed in parallel.

1) Consider the biggest square array.

2) Extract two interconnected triangular subarrays from each square array.

3) Build the minimum spanning tree in each triangular subarray. Two spanning trees are built in parallel for each subarray S_a one for each triangular array. The algorithm used is essential to reduce the execution time of the overall reconfiguration algorithm. In [1] the fastest algorithms for minimum spanning tree generation are described. The best running time achievable

without lack of generality is O(|E|loglog|V|) but for planar graphs O(|V|) can be achieved. The result is a connected square array in each subarray.

4) Complete connections. Complete connectivity of the array can now be provided, by visiting the tree from the root, and adding to each element that requires a connection its nearest neighbor belonging to the lower level and requiring a connection, in order to create the shortest circuit.

RESULTS

The reconfiguration algorithm here proposed is a unified approach to overcome two types of faults: randomly distributed faults and cluster fault distribution. Random fault distribution, which is suitable for run-time faults in VLSI/WSI arrays, assumes that the probability that a processing element is defective is independent from the status of its neighbors. The minimum spanning tree based algorithm allows externally host-driven reconfiguration of the array in presence of random faults. To establish interconnections between logically adjacent elements after reconfiguration 3*3 switched busses are adopted ([2]). The upper bound on wire lenght is given by the longest vertical or horizontal chain of adjacent faulty elements [3].

Cluster fault distribution represents large defective zones of the wafer in which irregular groups of neighboring elements and the links positioned among them are affected. This type of distribution is well suited for production-time faults in wafer scale arrays. The minimum spanning tree based algorithm allows static reconfiguration, at production-time, of the wafer scale array as shown in fig.1. The bound on maximum wire length, and thus on interconnection delays is given by half the perimeter plus one of the largest cluster of faults present in the array [3].

The algorithm has been evaluated by simulation on a relevant number of reconfiguration problems. A reconfiguration problem is defined by injecting faults in the elements of the array and considering the largest working array that can be obtained by applying the algorithm (harvesting).

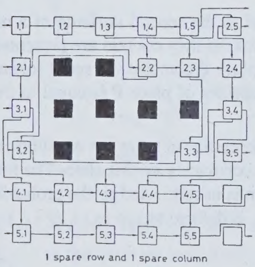


Fig. 1

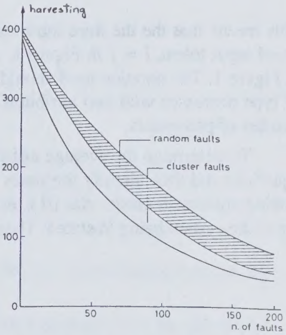


Fig. 2

Figure 2 shows the results for random and cluster distribution of faults: harvesting is computed as a function of the total number of faults inserted inside a 20*20 physical array.

REFERENCES

[1] Cheriton D., Tarjan R.E., "Finding Minimum Spanning Trees", SIAM J. of Computing, vol. 5, n. 4 (December 1976)

[2] Negrini R., Stefanelli R., "Comparative Evaluation of Space- and Time-Redundancy Aprroaches for WSI Processing Arrays", IFIP Workshop on Wafer Scale Integration, Grenoble, France (March 1986)

[3] Lombardi F., Sciuto D., "Reconfiguration and Delay Analysis of VLSI/WSI Arrays", in Department of Electrical and Computer Engineering, ed. University of Colorado, Boulder (July 1986)

Preliminary Results on Multiprocessor Modeling and Analysis Using Stochastic High Level Petri Nets

Dan Cristian Marinescu, Chuang Lin

Computer Sciences Department
Purdue University, West Lafayette, IN 47907, USA

Stochastic High Level Petri nets [SHLPNs] have been introduced in [1]. They are High Level Petri nets augmented with exponentially distributed firing times. We have concentrated our attention upon *homogeneous systems* which, when modeled using SHLPNs have subsets of *equivalent* states. Such states can be grouped together in such a way that the SHLPN model of the system with *compound markings* contains only one *compound state* for each group of *individual states* in the original SHLPN model. SHLPNs are isomorphic to continuous time Markov chains [2]. The compound marking concept represents a grouping of markings in the Petri net domain and it corresponds to state grouping in the Markov domain.

To simplify the analysis of a multiprocessor configuration used as a case study, we consider a homogeneous system. Moreover we model the behavior of each processor as follows: it switches between execution in a private domain, when its operands come from private memory and a common domain when its operands are in one of the common memory modules [3]. Figure 1 presents a Stochastic High Level Petri net model of a multiprocessor system. Though the graph representing the model is invariant to the system size, the state space of the SHLPN clearly depends upon the actual number of: processors, n, common memory modules, q, and busses, r. For our example we assume: n=5, q=3 and r=2, a configuration with five processors, three common memory modules, and two busses.

A token has a number of attributes, the first attribute being its type, We recognize three different types: p - processor, m - common memory, b -bus. The second attribute of a token is its *identity, id.* The id attribute is a positive integer with values depending upon the number of objects of a given type. In our example when type=p, id takes values in the set [1,5]. The tokens residing in place Q have a third attribute, the id of the common memory module they are going to refer next. The general format of a token is: <type, id, i, j,...>. The function *Attr* selects a given attribute of a token. For example the predicate associated with transition G is:

$$Attr\ (p, 3) = Attr\ (m, 2) \tag{1}$$

This means that the the third attribute of a p-typed input token should be equal to the second attribute of a m-typed input token, $i = j$ in Figure 1. The meaning of different places and the tokens they contain are presented in Figure 1. The notation used should be interpreted in the following way: the place P contains the set of tokens of type processor with two attributes, <p, i>, with $i \in [1,5]$. The maximum capacity of place P is equal to the number of processors.

To determine the average utilization of different system resources it is necessary to solve the equilibrium equations and then identify the states when each resource is idle as well as the occupancy of that state. The following notation is used: $size\ [B\ ]_i$ is the occupancy of place B when the system is in state i and p_i the probability of the system being in state i. Then the average utilization of a processor, η_p is defined as:

$$\eta_p = 1 - \sum_{i \in S} \frac{p_i \times size\ [\ Q\ ]_i}{n} \tag{2}$$

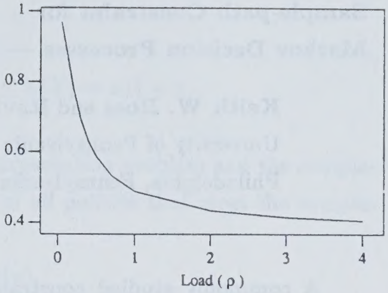
Figure 2 presents the processor utilization function of the load for common resources defined as $\rho = \frac{\lambda_1}{\lambda_3}$. The number of original states of the model is very high, larger than 500, and we have reduced the model to only 51 states.

References

[1] Chuang Lin and Dan Cristian Marinescu *"Stochastic High Level Petri Nets and Applications"* Computer Science Department, Purdue University, Technical Report CSD-TR-613, July 1986

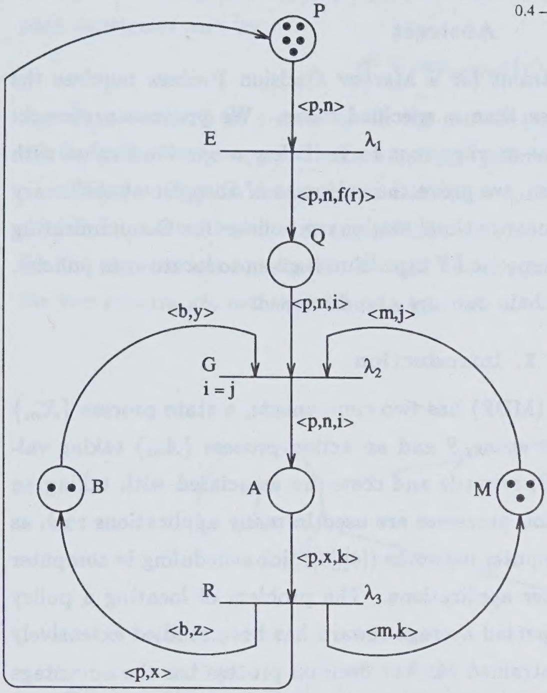
[2] M.K. Molloy, *"Performance Analysis Using Stochastic Petri Nets,"* IEEE Trans. Comp. C-31, 9, PP. 913-917, Sep. 1982

[3] M.A. Marsan, G. Conte and G. Balbo, *"A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor systems,"* ACM Trans. Comp. Systems, Vol. **2**, No. 2, PP. 93-122, May 1984



Load(ρ)	Utilization(η_p)
0.050000001	0.970678091
0.100000001	0.911530435
0.200000003	0.785639346
0.300000012	0.693749726
0.400000006	0.631783068
0.500000000	0.588843763
0.699999988	0.534519076
1.000000000	0.490343213
2.000000000	0.435558975

Figure 2. The utilization of a processor versus load



PLACE AND TRANSITION SIGNIFICANCE

P	is the "private memory" place. When the place P holds tokens, the corresponding processors are active in their own memory.
Q	is the "queuing" place. When the place Q holds tokens, the corresponding processors are queued for common memory.
A	is the "accessing" place. When the place A holds tokens, the corresponding processors are accessing common memory.
M	is the "idle common memory" place. When the place M holds tokens, the corresponding common memories are idle.
B	is the "available bus" place. When the place B holds tokens, the corresponding busses are available.
E	is the "end of activity" transition. When it fires, a processor ends its activity in its private memory.
G	is the "getting common memory" transition. The transition is enabled when the common memory module and a bus are free. The transition firing time is related to $1/\lambda_2$.
R	is the "releasing common memory" transition. After accessing common memory, the processor releases the common memory and the bus and returns to the private memory execution.

Figure 1. Modeling of the multiprocessor system with a SHLPN

Sample-path Constraint for Markov Decision Processes — A New Criterion

Keith W. Ross and Ravi Varadarajan
University of Pennsylvania
Philadelphia, Pennsylvania.

Abstract

A commonly studied constraint for a Markov Decision Process requires the expected average cost to be less than a specified value. We propose a *stronger* criterion which requires the time-average cost to be below a specified value with probability one. For this problem, we prove the existence of an optimal stationary policy for Unichain MDPs and near-optimal stationary policies for Communicating and Deterministic MDPs. A parametric LP algorithm is given to locate such policies. Extensions to the general multichain case are also discussed.

1. Introduction

A Markov decision process (MDP) has two components, a state process $\{X_m\}$ taking values from a finite state space S and an action process $\{A_m\}$ taking values from a finite action space B; rewards and costs are associated with taking an action in a state. Markov decision processes are used in many applications such as inventory control, routing in computer networks ([4],[9]), job scheduling in computer systems ([1]) and other computer applications. The problem of locating a policy that maximizes the long-run expected average reward has been studied extensively in the literature ([3],[6]). A constrained Markov decision process has the advantage that multiple cost or reward objectives can be considered for optimization. A commonly studied constrained Markov decision problem ([3],[6]) is to find a policy u that maximizes the average expected reward

$$R(u) = \liminf_{n\to\infty} \frac{1}{n} \sum_{m=1}^{n} E_u\big[r(X_m, A_m)|X_1 = x_1\big]$$

subject to a constraint on the average expected cost

$$K(u) = \limsup_{n\to\infty} \frac{1}{n} \sum_{m=1}^{n} E_u\big[c(X_m, A_m)|X_1 = x_1\big] \le \alpha.$$

In this paper, we define a new constrained Markov decision problem of locating a policy u that maximizes the expected average reward

$$\phi(u) = E_u\big[\liminf_{n\to\infty} \frac{1}{n} \sum_{m=1}^{n} r(X_m, A_m)|X_1 = x_1\big]$$

subject to the sample-path constraint

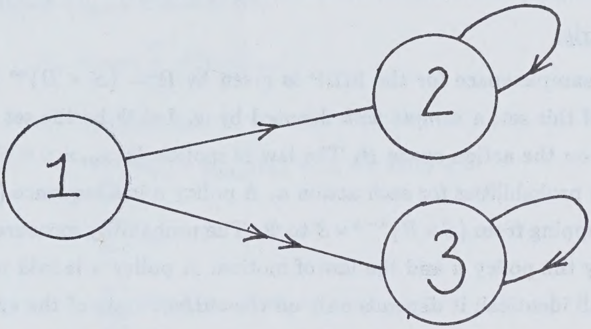
$$P_u\Big(\limsup_{n\to\infty}\frac{1}{n}\sum_{m=1}^{n}c(X_m,A_m)\le\alpha|X_1=x_1\Big)=1.$$

We will refer to the above two problems as the expectation problem and the sample-path problem respectively. Let U_0 be the class of all policies that meet the sample-path constraint and let

$$\phi^* = sup_{u\in U_0}\phi(u).$$

A policy is said to be *ϵ-optimal* for the sample-path problem, if $\phi(u)\ge\phi^*-\epsilon$ for $\epsilon\ge 0$. In the same way, we define ϵ-optimal policies for the expectation problem.

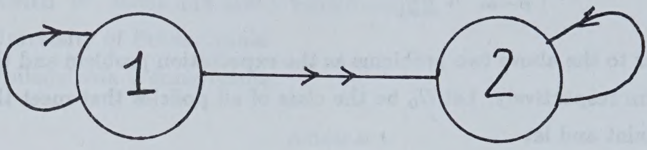
The sample-path constraint is a stronger constraint than the expectation constraint; by this we mean that any policy satisfying the sample-path constraint satisfies the expectation constraint. We will demonstrate by the following example that the two criteria are not equivalent in general.



$c(2,1)=r(2,1)=3$; $c(3,1)=r(3,1)=5$; $\alpha=4$ and the initial state is 1. In this example, for each state-action pair there is an unique next state; the actions are labeled by the arrowheads on the edges. In both expectation and sample-path problems, there exists an optimal stationary policy but the optimal values are not the same. In the sample path case, only the policy that brings the system to state 2, meets the constraint and the optimal value is equal to 3. In the expected average case however, the optimal stationary policy is the one that chooses actions 1 and 2 at state 1, each with probability 0.5; the optimal value is equal to 4.

For the expectation problem, Hordijk and Kallenberg [5] studied the structure of optimal policies for MDPs with general recurrent structures.

In this case, there does not always exist an optimal stationary policy, or even an ϵ-optimal stationary policy as can be seen from the following example:



$r(1,1) = c(1,1) = 0$; $r(1,2) = c(1,2) = 0$; $r(2,1) = c(2,1) = 1$; $\alpha = \frac{1}{2}$; the initial state is 1. For the expectation criterion, all feasible stationary policies earn zero expected reward. But consider the Markov policy which at $t = 1$ chooses actions 1 and 2, each with probability $\frac{1}{2}$, and for $t \geq 2$ always chooses action 1; this policy is feasible and earns an expected reward of $\frac{1}{2}$. Thus for the expectation criterion, there does not exist an optimal or even an ϵ-optimal stationary policy for $0 \leq \epsilon < \frac{1}{2}$. For the sample path criterion, there is an optimal stationary policy.

Preliminaries

The sample space for the MDP is given by $\Omega = \{S \times B\}^\infty$ and we call each element of this set, a *sample path* denoted by ω. Let Ψ be the set of all probability measures on the action space B. The law of motion $\{p_{xay}, x, y \in S, a \in B\}$ specifies transition probabilities for each action a. A policy u is a sequence $(u_1, u_2, \ldots)$ where u_k is a mapping from $\{S \times B\}^{k-1} \times S$ to Ψ. The probability measure P_u is completely defined by the policy u and the law of motion. A policy u is said to be *stationary* if u_k's are all identical; it depends only on the current state of the system. We denote a stationary policy f by a vector $\{f_{xa}, x \in S, a \in B\}$. A stationary policy f is said to be *pure* if for each state x, there is an unique action a such that $f_{xa} = 1$.

Under any stationary policy f, the state process $\{X_m\}$ is a Markov chain with transition matrix $P(f)$ given by

$$P_{xy}(f) = \sum_a p_{xay} f_{xa}.$$

A transition matrix $P(f)$ is *unichain* if it has at most one recurrent class; the unique steady-state vector for the underlying Markov process is given by $\pi(f)$. It is a well-known fact (see [2]) that if for a stationary policy f the transition matrix $P(f)$ is unichain, then P_f-almost surely

$$\pi_x(f) = \lim_n \frac{1}{n} \sum_{m=1}^{n} 1(X_m = x), \text{ for all } x \in S.$$

Here, $1(A)$ is the indicator function of the set A.

In this paper, we study the sample-path problem for the three natural classes of MDPs namely *unichain, communicating* and *deterministic MDPs*. A MDP is *unichain* if $P(g)$ is unichain for each pure policy g. A MDP is *communicating* if for each ordered pair of states (x, y), there exists a pure policy g such that y is accessible from x under $P(g)$. A MDP is *deterministic* if for every state x and action a there is an unique state y such that $p_{xay} = 1$.

2. Optimal Stationary policies

For the theorems presented in this and the subsequent sections, we either give an outline of the proof or omit the proof. For a complete discussion and for details of the proofs, we refer the reader to [11].

In this section, we present conditions for the existence of optimal stationary policies for the sample-path problem. For each fixed $\eta \geq 0$, we define the following LP Q_η, with variables $\{z_{xa}, x \in S, a \in B\}$:

$$T_\eta = \max \sum_{x,a} r(x,a) z_{xa}$$

$$\begin{aligned}
&\text{subject to} \\
&\sum_{x,a} (\delta_{xy} - p_{xay}) z_{xa} = 0, \text{ for all } y \in S \\
&\sum_{x,a} z_{xa} = 1 \\
&\sum_{x,a} c(x,a) z_{xa} \leq \alpha \\
&z_{xa} \geq \eta, \text{ for all } x \in S,\ a \in B.
\end{aligned}$$

For a given solution, let $z_x = \sum_a z_{xa}$ and $I_z = \{x \in S : z_x > 0\}$ and let f^z be a stationary policy satisfying

$$f^z_{xa} = \begin{cases} z_{xa}/z_x, & \text{if } x \in I_z; \\ \text{arbitrarily defined,} & \text{otherwise.} \end{cases}$$

The following theorem gives an upper bound for the expected average reward earned by a sample-path feasible policy. The proof makes use of the following fact from [7]:

Fact 1: For every policy u, we have P_u-almost surely

$$\lim_n \frac{1}{n} \sum_{m=2}^{n} [1(X_m = y) - \sum_{x,a} 1(X_{m-1} = x, A_{m-1} = a) p_{xay}] = 0$$

for all $y \in S$.

Theorem 1. *If there exists a sample-path feasible policy u, then Q_0 has a feasible solution and $\phi^* \leq T_0$.*

Proof (outline): We fix a feasible policy u and a sample path $\omega = (x_1, a_1, \ldots)$ satisfying the following:

$$\limsup_{n \to \infty} \frac{1}{n} \sum_{m=1}^{n} c(x_m, a_m) \leq \alpha \tag{1}$$

$$\lim_{n} \frac{1}{n} \sum_{m=2}^{n} [1(X_m = y) - \sum_{x,a} 1(X_{m-1} = x, A_{m-1} = a) p_{xay}] = 0, \tag{2}$$

for all $y \in S$. Let

$$\psi_n = \frac{1}{n} \sum_{m=1}^{n} r(x_m, a_m) \text{ and } \lambda_n = \frac{1}{n} \sum_{m=1}^{n} c(x_m, a_m).$$

Because u is a feasible policy and because of fact 1, it suffices to show that Q_0 is nonempty and that

$$\liminf_{n} \psi_n \leq T_0.$$

It is possible to pick a subsequence $\{n_k\}$ such that the following averages converge along $\{n_k\}$.

$$\lim_{k} \frac{1}{n_k} \sum_{m=1}^{n_k} 1(x_m = x, a_m = a) = w_{xa}, \; x \in S \text{ and } a \in B. \tag{3}$$

From (1),(2)and (3), we can show that $\{w_{xa}\}$ is a feasible solution to Q_0 and that

$$T_0 \geq \sum_{x,a} r(x,a) w_{xa} \geq \liminf_{n} \psi_n$$

Theorem 2. *In the case of unichain and communicating MDPs, there exists a sample-path feasible stationary policy if and only if Q_0 has a feasible solution*

Proof (outline): By theorem 1 it suffices to show the if part. For this, we assume that $\{z_{xa}\}$ is a feasible solution to Q_0 and derive a sample-path feasible policy $\tilde{f}$ from the stationary policy f^z as defined earlier. For the unichain MDP, we can show that f^z is a sample-path feasible policy and hence we let $\tilde{f} = f^z$. The remaining part of the proof is for the communicating case. Let $R_1, R_2, \ldots, R_m$ be the recurrent classes associated with $P(f^z)$. For each $1 \leq k \leq m$, we denote d_k to be the expected average cost under policy f^z, given that the initial state belongs to

R_k. Let l be such that d_l is the minimum value among all d_k's. Now we define $\tilde{f}$ as follows:

$$\tilde{f}_{xa} = \begin{cases} f^z_{xa}, & \text{if } x \in R_l; \\ \text{defined such that } \tilde{f}_{xa} > 0 \text{ for all } a \in B, & \text{if } x \notin R_l. \end{cases}$$

Now $\tilde{f}$ drives the state process into the class R_l. From the fact that $d_l \leq \alpha$, we then show that $\tilde{f}$ is a sample-path feasible policy.

Theorem 3. *Let $\{z_{xa}\}$ be an optimal solution to the LP Q_0. If the transition matrix of f_z has at most one ergodic class then f_z is an optimal policy.*

We omit the proof here. We note that in the case of unichain MDPs, f^z has at most recurrent class for every solution $\{z_{xa}\}$.

Thus combining the above theorems, we conclude that for the unichain MDPs, an optimal stationary policy for the sample-path criterion always exists provided that there exists a feasible policy; this policy can be obtained by solving Q_0. From [3], it follows then that both the sample-path and expectation criteria are identical in the case of unichain MDPs.

3. Near-optimal stationary policies

We give the theorems regarding communicating MDPs; the results regarding deterministic MDPs follow easily from these theorems. Let Z be the set of feasible solutions $\{z_{xa}\}$ to LP Q_0 such that $z_{xa} > 0$ for all $x \in S$ and $a \in B$.

Theorem 4. *Suppose that the MDP is communicating and Z is nonempty. Then for each $\epsilon > 0$, there exists an ϵ-optimal stationary policy for the sample-path criterion.*

Proof (outline): If Z is nonempty, then there is a $\varsigma > 0$ such that for each $0 \leq \eta < \varsigma$ there is a feasible solution for the LP Q_η. From an optimal solution $\{z^\eta_{xa}\}$ to the LP Q_η, we define a stationary policy f^η defined by

$$f^\eta_{xa} = \frac{z^\eta_{xa}}{z^\eta_x}, \; x \in S, a \in B \, .$$

The Markov chain under the transition matrix $P(f^\eta)$ is irreducible. By theorem 1, it suffices to show that f^η is feasible for all $0 < \eta < \varsigma$ and that

$$\lim_{\eta \downarrow 0} \phi(f^\eta) = T_0.$$

We prove this using the facts that $P(f^\eta)$ is irreducible for all $0 < \eta < \varsigma$ and that Z is nonempty.

The following theorem implies that for all practical purposes we may consider Z to be nonempty.

Theorem 5. *Suppose there exists a feasible policy u and a $\nu > 0$ such that*

$$P_u(\limsup_n \frac{1}{n} \sum_{m=1}^{n} c(X_m, A_m) \leq \alpha - \nu) = 1.$$

Then Z is nonempty.

We omit the proof here. Now we address the degenerate case of Z being empty.

Theorem 6. *Suppose that the MDP is communicating and that Z is empty. Let $\{z_{xa}\}$ be an optimum extreme point to the LP Q_0. Then the stationary policy f_z as defined before, is an optimal policy for the sample-path criterion.*

Proof (outline): We first show that if $P(f^z)$ has m recurrent classes then $\{z_{xa}\}$ can be expressed as a convex combination of m vectors. If Z is empty then we prove using theorem 5 that all these m vectors are feasible solutions to Q_0. Since $\{z_{xa}\}$ is an extreme point, m must be equal to 1, i.e., $P(f^z)$ is unichain. Hence by theorem 3, f^z is an optimal stationary policy.

Algorithm for Communicating MDPs

Thus the procedure for locating an optimal or a near optimal stationary policy in communicating MDPs is as follows:

1. Solve the LP Q_0. If Q_0 is infeasible, then there is no feasible sample-path policy.
2. Let $\{z_{xa}\}$ be an optimal extreme point to the LP Q_0. If f_z as defined before has at most one recurrent class then f_z is an optimal stationary policy; otherwise go to step 3.
3. Solve the parametric LP $(Q_\eta, \eta \geq 0)$ over some interval $[0, \varsigma]$ beginning with $\eta = 0$. Then from the optimal solutions we derive an ϵ-optimal stationary policy for ϵ as small as desired.

Remark: We see that for the unichain and communicating cases, the optimal (resp. ϵ-optimal) policies obtained from the LPs are uniformly optimal (resp. ϵ-optimal) over all initial states. However, this need not be true for deterministic MDPs.

4. Deterministic MDPs

In this section, we discuss briefly how we can construct ϵ-optimal stationary policies in the case of deterministic MDPs. The proofs for most of the results follow directly from the results for the communicating case.

First we partition the state space into equivalence classes such that if x and y belong to the same class then x and y *communicate* under some stationary policy. These classes are exactly the strongly connected components of the directed graph $G = (V, E)$ where V is the set of vertices corresponding to the states and $E = \{(x, y) | p_{xay} = 1 \text{ for some } a\}$. Let these classes be $C_1, C_2, \ldots, C_m$. For each $k = 1, 2, \ldots, m$ and for each $x \in C_k$, we define

$$B_x = \{a \in B | p_{xay} = 0 \text{ for all } y \notin C_k\}.$$

If $x \in C_k$ then B_x is the set of actions which keep the state process within C_k. Note that if B_x is empty for a state x, then x is transient under all stationary policies and hence x is a singleton class. Let Γ be the set of indices of classes C_k such that B_x is nonempty for all $x \in C_k$; the state process has to end up in one of these classes.

For each $k \in \Gamma$, we have a communicating MDP with state space C_k and state-dependent action space B_x for each $x \in C_k$. For each such MDP, we define LPs similar to Q_η as in section 2. For each k, let T_0^k be the optimal value of LP Q_0^k and let l be such that T_0^l is the maximum among all T_0^k's. We create an ϵ-optimal stationary policy f^ϵ as follows : when in state $x \notin C_l$, f^ϵ drives the process into a state in C_l; when in state $x \in C_l$, it follows an ϵ-optimal stationary policy for the communicating MDP of class C_l.

5. Conclusions

We have shown how to derive optimal or near-optimal stationary policies for the sample-path problem in the case of unichain,communicating and deterministic MDPs. For the deterministic MDPs, we have taken the approach of decomposing the state space into communicating MDPs. We have extended this decomposition theory to solve the problem for the general multichain case; this is the subject of a companion paper [10].

References

[1] Chow,Y.C.,and Kohler,W.H., "Models for dynamic load balancing in a heterogeneous multiple processor system," IEEE trans. Computers, vol.c-28,no.5, May 1979.

[2] Cinlar,E.,*Introduction to Stochastic Processes*, Prentice-Hall,Englewood Cliffs, NJ, 1975.

[3] Derman,C.,*Finite State Markovian Decision Processes*, Academic Press, New York, 1970.

[4] Foschini,G.J., "On heavy traffic diffusion analysis and Dynamic routing in packet-switched networks," *Computer Performance*, Eds. K.M. Chandy and M.Reiser,pp.499-513,1977.

[5] Hordijk,A.,and Kallenberg,L.C.M.,"Constrained Undiscounted Stochastic Dynamic Programming," Mathematics of Operations Research,vol.9, pp.276-289, May 1984.

[6] Kallenberg,L.C.M., *Linear Programming and Finite Markovian Control Problems*, Mathematical Centre Tracts 148, Amsterdam, 1983.

[7] Loeve,M., *Probability Theory, Vol. 2*, Springer-Verlag, New York, 1978.

[8] Murty,K.G., *Linear programming*, John Wiley and Sons, New York, 1983.

[9] Nain,P.,and Ross,K.W., "Optimal Multiplexing of Heterogeneous Traffic with Hard Constraint," Performance Evaluation Review, Vol.14, no.1, 1986.

[10] Ross.K.W., and Varadarajan,R., "The decomposition of time-average MDPs: theory, algorithms and applications," submitted to SIAM J. of Control and Optimization.

[11] Ross,K.W., and Varadarajan,R., "Markov Decision Processes with Sample Path Constraints: Unichains, Communicating and Deterministic Cases," submitted to Operations Research, 1986.

OPTIMAL PRIORITY ASSIGNMENT WITH GENERAL CONSTRAINTS

EITAN ALTMAN and ADAM SHWARTZ [1]
Electrical Engineering, Technion, Haifa, ISRAEL

INTRODUCTION

We consider a G/M/1/∞ discrete time queueing system with K classes of customers. During a time slot (t,t+1) a customer from any class $1 \le i \le K$ may be served. If served, with probability μ_i it leaves its respective queue; otherwise it remains. A policy determines which customer class is served at each slot. Arrivals are independent from slot to slot, and form a renewal sequence with means λ_k.

In section 2 we solve an optimization problem with a constraint of the form of a linear combination of the average line lengths of K customers' classes, using a randomized version of the μc rule [1,3]. In section 3 we introduce "time sharing" policies, and solve the optimization problem under several constraints. This method provides *explicit* solutions [4], and is also applicable to the problems of [1,3]; it is not clear whether there is any other optimal policy for the multiple constraint problem.

OPTIMIZATION UNDER ONE CONSTRAINT

Define $N_k(s) :=$ number of class-k customers at time s, $N(s) := \{N_1(s), \dots, N_k(s)\}$, u:= policy used. Given the positive c_i, d_i and V, we solve:

(2.1) minimize a linear combination $C_m(u)$ of the average line lengths, subject to a constraint on another linear combination of the average line lengths $D_m(u) \le V$, where :

$$C_m(u) := \overline{\lim_{t\to\infty}} \frac{1}{t} E_u \left[\sum_{s=1}^{t} \sum_{k=1}^{K} c_k N_k(s) \mid N(1)=m \right],\ D_m(u) := \overline{\lim_{t\to\infty}} \frac{1}{t} E_u \left[\sum_{s=1}^{t} \sum_{k=1}^{K} d_k N_k(s) \mid N(1)=m \right] \quad (1)$$

Consider the unconstrained problem with cost parametrized by $0 \le \theta \le 1$, similar to the Lagrange formulation in [1,3];

$$J_m^\theta(u) := \overline{\lim_{t\to\infty}} \frac{1}{t} E_u \left[\sum_{s=1}^{t} \left(\sum_{k=1}^{K} \theta c_k N_k(s) + (1-\theta) d_k N_k(s) \right) \mid N(1)=m \right] \quad (2)$$

where the optimal cost is $J_m^\theta := \inf_u J_m^\theta(u)$. If $\rho := \sum_{k=1}^{K} \frac{\lambda_k}{\mu_k} < 1$, then for any non idling (empty queues are not served unless the system is empty) stationary policy f, the costs $C_m(f)$, $D_m(f)$ and $J_m(f)$ are independent of the initial state m (which we then omit), and limits in (1) and (2) exist, so we obtain: $J^\theta(f) = \theta C(f) + (1-\theta) D(f)$ [1,2]. Denote by G the class of K! non idling policies g_i obtained by all possible orderings of priorities among the K queues. For each θ there exists a policy g(θ) in G that satisfies the μc rule [1-2], hence is optimal for (2) [2]. Since $\tilde{c}_k(\theta) := d_k + \theta(c_k - d_k)$ are affine in θ, for each $i \ne j$ there is at most one $\theta := \theta_{ij}$ for which $\tilde{c}_i(\theta) = \tilde{c}_j(\theta)$ (ignore the case $d_i = d_j$ and $c_i = c_j$). Between two adjacent θ_{ij}, $g(\theta)$ remains constant. Hence, as θ varies from 0 to 1, we have $n \le K!$ policies $g_i \in G$, that change at θ_i's, where w.l.o.g. $0 = \theta_1 \le \cdots \le \theta_{n+1} = 1$, and g_i minimizes $J_m^\theta(u)$ for $\theta_i \le \theta \le \theta_{i+1}$. Define f_{ql} as the randomized policy that employs either g_{l-1} *or* g_l with probability q and $1-q$ respectively at each decision epoch. Then [1] f_{ql} minimizes $J_m^{\theta_l}(u)$ for any q and m and $D(f_{ql})$ is continuous in q. Define $I := \min\{i > 1 : D(g_i) \ge V\}$.

Theorem 1: If $D(g_1) \le V \le D(g_n)$ then f_{qI} is an optimal (randomized) policy for (2.1). If $D(g_1) > V$ there is no feasible policy for (2.1). If $D(g_n) < V$ then g_n is constrained optimal.
Proof: By definition, $D(g_{I-1}) \le V \le D(g_I)$, so by the continuity of $q \to D(f_{qI})$ there exists a q such that $D(f_{qI}) = V$. Since f_{qI} minimizes $J_m^{\theta_I}(u)$ it follows that, for any u and m :

$$J(f_{qI}) = \theta_I C(f_{qI}) + (1-\theta_I) D(f_{qI}) = \theta_I C(f_{qI}) + (1-\theta_I) V \le J_m^{\theta_I}(u) \le \theta_I C_m(u) + (1-\theta_I) D_m(u) \quad (3)$$

This implies that $C(f_{qI})$ is constrained optimal since, for any feasible policy u,

$$\theta_I [C(f_{qI}) - C_m(u)] \le (1-\theta_I)[D_m(u) - V] \le 0 \quad (4)$$

---

[1] Research supported in part by a United States - Israel Binational Science Foundation grant no. 85-00306.

If $D(g_1)>V$ there is no *feasible* policy, since g_1 minimizes $J_m^0(u):=D_m(u)$. If $D(g_n)<V$ then g_n meets the constraint and minimizes $C_m(u)$ (since $C_m(u)=J_m^1(u)$ and g_n is optimal for $J_m^1(u)$). ☐

The disadvantage of this randomization method is that there is no explicit expression for the bias factor q [1], and numerical methods must be used in order to implement the policy (but see [4]).

THE "TIME SHARING" APPROACH

With the notation of section 2, let $l:=K!$. Recall the recurrence properties, and that if u is stationary, C_m^i is independent of m. In this section we introduce "time sharing" policies and solve:

(3.1) Minimize $C_m^K(u):=$ average line length of customer K class, under K-1 constraints on the remaining line lengths $C_m^i(u)$, i.e. we impose: $C_m^i(u)\leq V^i$, $1\leq i\leq K-1$, where:

$$C_m^i(u):=\overline{\lim_{t\to\infty}}\frac{1}{t}E_u\left[\sum_{s=1}^{t}N_i(s)\mid N(1)=m\right]\quad\text{for } 1\leq i\leq K \tag{5}$$

A "time sharing" policy $\hat{\alpha}$ uses, between consecutive times the system is empty (a cycle), a constant policy g_i, so that the fraction of number of cycles during which the different g_i policies were used converges to $\alpha:=(\alpha_1,\cdots,\alpha_l)$. For example, if $\alpha:=(3/6, 2/6, 1/6, 0,\ldots,0)$ then g_1 is used 3 times, g_2 twice and g_3 once every 6 cycles, and other policies are not used. Using renewal arguments and the fact that the expected cycle duration τ when using g_i is finite and independent of i [2], it can be shown that $C^i(\hat{\alpha})=\sum_{j=1}^{l}\alpha_j C^i(g_j)$. Note that the $C^i(g_j)$'s have explicit expressions [1]. Thus our problem reduces to solving a linear programming problem in l unknown α_i's :

$$\textit{Minimize}\quad C^K(\hat{\alpha})=\sum_{j=1}^{l}\alpha_j C^K(g_j),\qquad \textit{subject to} \tag{6}$$

$$C^i(\hat{\alpha})=\sum_{j=1}^{l}\alpha_j C^i(g_j)\leq V^i,\ 1\leq i\leq K-1\ ;\qquad \alpha_j\geq 0,\ 1\leq j\leq l\ ;\qquad \sum_{j=1}^{l}\alpha_j=1 \tag{7}$$

Theorem 2: If there is a solution to (6-7), then the policy $\hat{\alpha}$ is optimal for (3.1).
Proof: Denote by L the class of indices i for which $C^i(\hat{\alpha})=V^i$. One can show that the policy $\hat{\alpha}$ is also optimal for problem (6-7) restricting in (7) to $i\in L$. As in section 2, introduce the unconstrained problem of minimizing $J_m(u)$, where :

$$J_m(u):=\overline{\lim_{t\to\infty}}\frac{1}{t}E_u\left[\sum_{s=1}^{t}\left(\sum_{j\in L}\frac{1}{\mu_j}N_j(s)+\frac{1}{\mu_K}N_K(s)\right)\mid N(1)=m\right] \tag{8}$$

According to the μc rule [1-2], any non idling policy is optimal for $J_m(u)$ (because the appropriate "μc"s equal 1 for all j). Using (7), for any m and feasible u :

$$J_m(\hat{\alpha})=\sum_{j\in L}\frac{1}{\mu_j}V^j+\frac{C^K}{\mu_K}(\hat{\alpha})\leq J_m(u)\leq\sum_{j\in L}\frac{1}{\mu_j}C_m^j(u)+\frac{C^K(u)}{\mu_K}\leq\sum_{j\in L}\frac{1}{\mu_j}V^j+\frac{C^K(u)}{\mu_K} \tag{9}$$

Therefore $C^K(\hat{\alpha})\leq C^K(u)$. ☐

It can be shown that the converse of theorem 2 is also valid i.e. if there is no "time sharing" solution to (6-7) then there does not exist any solution to (3.1). The "time sharing" policy has the advantage that α can be explicitly calculated, rather than computed numerically, as for the randomized policy.

REFERENCES

[1] Nain, P. and K. W. Ross, "Optimal Priority Assignment with hard Constraint," IEEE Trans. on Automatic Control Vol. AC-31 No. 10, October 1986.

[2] Baras, J. S., A. J. Dorsey, and A. M. Makowski, "Discrete time competing queues with geometric service requirements: stability, parameter estimation and adaptive control," Report no. SRR 84-17, Electrical Engineering, University of Maryland, College Park, Maryland, 1984.

[3] F. J. Beutler and K. W. Ross, "Optimal policies for controlled Markov chains with a constraint", Math. Anal. Appl. 112, pp. 236-252, 1985.

[4] Shwartz A. and A. M. Makowski, "An Optimal Adaptive Scheme for Two Competing Queues with Constraints", in Analysis & Opt. of Systems, edited by A. Bensoussan and J. L. Lions, Springer Verlag lecture notes in Control & Info. Scie. No. 83, 1986.

DYNAMIC SCHEDULING OF THREE QUEUES IN TANDEM

Rodolfo A. Milito
AT&T Bell Laboratories, Holmdel, N.J. 07733
Derya Cansever
Univ. of Massachusetts, Dept. of Electrical and Computer Engineering, Amherst, MASS. 01003

Abstract

Three queues in tandem compete for a single server. At each instant of time the cost is a weighted sum of the number of customers in the queues. The problem of finding the optimal control policy can be considered as a *dual* of the celebrated problem of dynamically scheduling a number of queues in parallel which share a single server. We analyze the conditions for optimality, and in particular we derive the optimal policy for the infinite horizon discounted version of the problem. In doing so, we find that the optimal policy is not always myopic, which contrasts with the well known "$c\mu$" rule for the *primal* problem.

1. INTRODUCTION

Considerable attention has been paid lately to the area of dynamic control of queueing systems. Policies which base their decisions on dynamic information about a system certainly outperform open loop policies. In some applications, the improvement in performance more than compensates for the extra complexity of the dynamic policy. Manufacturing systems and computer communication networks are prime candidates for applications of dynamic scheduling.

The by now classical papers of Klimov [1] ,[2], deal with the problem of dynamically scheduling a single server in a queueing network in order to minimize **the long run weighted average** of the number of customers in the system within the class of non-preemptive policies. An "index rule" is shown to be optimal. A number of queues in parallel sharing a single server, with a linear cost structure and under a long run average performance measure, is a particular important case of the problem addressed by Klimov. As such, it has been extensively studied [3]. The *"index rule"* takes in this case a simple, appealing form, known as the "$c\mu$" rule. Buyukkoc, Varaiya and Walrand supplied a very elegant proof of the optimality of the "$c\mu$" rule for **both the long run average and the discounted problems,** provided that the the service times are geometrically distributed [4], and preemption is allowed.

In this paper we address a problem which can be considered as a "dual" of the above: We have three queues in tandem which compete for a single server, and we consider a discounted performance cost.

This structure leads to interaction between the queues in the sense that the output of one queue is the input to the next one. Hence, our model does not enjoy the nice separation property that serving one queue does not affect the state of the others. Nevertheless, we show the optimality of a variant of the "$c\mu$" rule for a certain range of parameters of a network of queues in tandem.

In the limit, when the discount factor tends to one, this optimal policy converges to the optimal policy for the long run average version of the problem, as derived according to Klimov's *"index rule"*. Therefore, the optimal policies with and without preemption coincide when a long run average measure of the performance is used.

2. STATEMENT OF THE PROBLEM

We have an open network consisting of three queues in series competing for a single server. An external arrival stream of customers is received by the first queue. We assume that this stream is Poisson with rate λ. Whenever a customer in the first (second) queue completes service, it is immediately transferred to the second (third) queue. Customers leave the system after completing service in the third queue. The service time required by a customer in queue Q_i is exponentially distributed with parameter μ_i.

We assume that the buffers of the three queues are infinite, and consider the following performance measure:

$$J^{\alpha}_{\pi}\left(x, T\right) = E_{\pi}\left\{\int_{0}^{T}\left[c_1\, x_1(t) + c_2\, x_2(t) + c_3\, x_3(t)\right] e^{-\alpha t} dt \,/ x(0) = x\right\}, \tag{2.1}$$

where:

$c_1 \geq c_2 \geq c_3 \geq 0,$

$x_i(t)$ = *number of customers in queue i at time t*,

$x = (x_1, x_2, x_3)$

$\alpha > 0$ (*discount rate*),

and π denotes a policy which assigns the server to one of the three queues at each instant of time.

Our problem is to find an optimal policy π^* which minimizes this cost functional within the class of non-anticipative, preemptive[1] , perfect state information policies.

A standard uniformization technique (see [6] , [7] , [8]) is used to obtain an equivalent discrete time formulation of the problem. In this formulation decisions are only taken at transition epochs (*either real or potential ones*), and the "total event rate" is

$$r = \lambda + \mu_1 + \mu_2 + \mu_3 \tag{2.2}$$

The discount factor β, which is the discrete time counterpart of the discount rate α, is given by

$$\beta = \frac{r}{r+\alpha}. \tag{2.3}$$

We now define four operators which describe the operations of arrival, departure and transfer of a customer from one queue to the next one:

$$\mathbf{A}x = (x_1 + 1\,, x_2\,, x_3\,) \tag{2.4}$$

$$\mathbf{D}x = \begin{cases} (x_1\,, x_2\,, x_3 - 1\,) & \text{if } x_3 > 0 \\ (x_1\,, x_2\,, x_3\,) & \text{if } x_3 = 0 \end{cases} \tag{2.5}$$

$$\mathbf{R}_{12}x = \begin{cases} (x_1 - 1\,, x_2 + 1\,, x_3\,) & \text{if } x_1 > 0 \\ (x_1\,, x_2\,, x_3\,) & \text{if } x_1 = 0 \end{cases} \tag{2.6}$$

$$\mathbf{R}_{23}x = \begin{cases} (x_1\,, x_2 - 1\,, x_3 + 1\,) & \text{if } x_2 > 0 \\ (x_1\,, x_2\,, x_3\,) & \text{if } x_2 = 0 \end{cases} \tag{2.7}$$

Let $V_n(x)$ denote the optimal cost to go for the n remaining epochs. Then, the Dynamic Programming equation which characterizes the problem is given by:

$$V_0(x) = 0 \tag{2.8a}$$

$$V_{n+1}(x) = c_1 x_1 + c_2 x_2 + c_3 x_3 + \beta\lambda V_n(\mathbf{A}x)$$

---

[1] Either preemptive resume or preemptive repeat with resampling[5] (which are equivalent in this case since the service time distribution is exponential).

$$+\beta \min \Big\{ \mu_1 V_n(\mathbf{R}_{12}x) + (\mu_2 + \mu_3)V_n(x) \,,$$

$$\mu_2 V_n(\mathbf{R}_{23}x) + (\mu_1 + \mu_3)V_n(x) \,,\ \mu_3 V_n(\mathbf{D}x) + (\mu_1 + \mu_2)V_n(x) \Big\} \tag{2.8b}$$

The DP equation leads immediately to the following decision rules:

$$\mu_1\Big[V_n(x) - V_n(\mathbf{R}_{12}x)\Big] \underset{\mathbf{Q}_2}{\overset{\mathbf{Q}_1}{\gtrless}} \mu_2\Big[V_n(x) - V_n(\mathbf{R}_{23}x)\Big] \tag{2.9}$$

$$\mu_1\Big[V_n(x) - V_n(\mathbf{R}_{12}x)\Big] \underset{\mathbf{Q}_3}{\overset{\mathbf{Q}_1}{\gtrless}} \mu_3\Big[V_n(x) - V_n(\mathbf{D}x)\Big] \tag{2.10}$$

$$\mu_2\Big[V_n(x) - V_n(\mathbf{R}_{23}x)\Big] \underset{\mathbf{Q}_3}{\overset{\mathbf{Q}_2}{\gtrless}} \mu_3\Big[V_n(x) - V_n(\mathbf{D}x)\Big] \tag{2.11}$$

where the notation $\underset{\mathbf{Q}_j}{\overset{\mathbf{Q}_i}{\gtrless}}$ means that the optimal server prefers to serve $\mathbf{Q}_i$ ($\mathbf{Q}_j$) when the inequality > (<) prevails.

Define:

$$\Delta_1^n = \mu_1\Big[V_n(x) - V_n(\mathbf{R}_{12}x)\Big], \tag{2.12}$$

$$\Delta_2^n = \mu_2\Big[V_n(x) - V_n(\mathbf{R}_{23}x)\Big] \tag{2.13}$$

$$\Delta_3^n = \mu_3\Big[V_n(x) - V_n(\mathbf{D}x)\Big], \tag{2.14}$$

and note that Δ_j^n represents, modulo a constant, the expected decrease in the cost to go when $\mathbf{Q}_j$ is served.

We can now introduce the concept of *dominance*, first in a local and then in a global sense:

- **$\mathbf{Q}_j$ dominates $\mathbf{Q}_i$ at epoch n and state x iff:**

$$\Delta_j^n(x) \geq \Delta_i^n(x)$$

- $\mathbf{Q}_j > \mathbf{Q}_i$, to be read " **$\mathbf{Q}_j$ dominates $\mathbf{Q}_i$** " (without further qualifications), iff:

$$\Delta_j^n(x) \geq \Delta_i^n(x)$$

holds for all epochs n and all states x such that $x_j > 0$.

3. SOME STRUCTURAL PROPERTIES OF QUEUES IN TANDEM

Before focusing on the specific problem of three queues in tandem we will derive some general results for N of queues in series, where N is any positive integer. The problem formulation, as well as the notation just introduced, extends from the three to the N case in an obvious way.

The condition that the weighing cost c_i should be non-increasing in i is enforced to avoid certain technicalities with short planning horizons.

Proposition 1:

Let i, j be nonnegative integers, and suppose the following holds:

— **Condition C1:**

$$\Big(c_{i+1+j} - c_{i+2+j}\Big)\mu_{i+1+j} > \Big(c_i - c_{i+1}\Big)\mu_i$$

When $j > 0$ further assume that:

— **Condition C2:**

Either

$$\mathbf{Q}_i > \mathbf{Q}_{i+1},$$

or

$$\mathbf{Q}_{i+1+j} > \mathbf{Q}_{i+1}.$$

Then:

—

$$\mathbf{Q}_{i+1+j} > \mathbf{Q}_i$$

Proof:

We use induction. Let n denote the number of stages to go, and π an optimal policy as prescribed by the above Proposition. if the proposition holds.

- The claim trivially holds for $n = 1$ and it also holds for $n = 2$ (the myopic decision is optimal when there is only one stage to go).
- Let us assume it holds for some generic $n > 2$.
- We need to prove that the claim holds for $n+1$. In order to do so we will use contradiction, which will be unveiled via an interchange argument.
- Suppose that at epoch $n+1$, i.e., when there are $n+1$ stages to go, both $\mathbf{Q}_i$ and $\mathbf{Q}_{i+1+j}$ have customers. By way of contradiction let us assume it is optimal to serve $\mathbf{Q}_i$ at this stage. Let $\bar{\pi}$ denote this new policy, which by the induction hypothesis is identical to π in the interval $\left[n\ ,1\right]$. Let us say that under $\bar{\pi}$, $\mathbf{Q}_{i+1+j}$ is served σ stages later, where σ is a random but finite number (since the decision at epoch 1 is arbitrary, we can claim without loss of generality that $0 < \sigma \leq n$).
- In order to show that $\bar{\pi}$ cannot be optimal we define a new policy $\hat{\pi}$ by interchanging the decisions that $\bar{\pi}$ makes at epochs $n+1$ and $n+1-\sigma$. It should be noted that under policy $\bar{\pi}$ (identical to π except at epoch $n + 1$), $\mathbf{Q}_i$ is not served during the interval $\left[n, n-\sigma\right]$. Furthermore, C2 guarantees that feasible decisions under $\bar{\pi}$ are feasible under $\hat{\pi}$ too. So, both policies take the same decisions at all epochs, except for the interchanged ones. Nothing is altered after epoch $n + 1 - \sigma$, so the relative savings expected by using policy $\hat{\pi}$ instead of $\bar{\pi}$ are given by:

$$\begin{aligned}
\Delta\left(\hat{\pi},\bar{\pi}\right) = \\
&= \left(c_{i+1+j} - c_{i+2+j}\right)\mu_{i+1+j}E\sum_{s=1}^{\sigma+1}\beta^s - \left(c_i - c_{i+1}\right)E\beta^{\sigma+1} \\
&- \left(c_i - c_{i+1}\right)E\sum_{s=1}^{\sigma+1}\beta^s - \left(c_{i+1+j} - c_{i+2+j}\right)\mu_{i+1+j}E\beta^{\sigma+1} \\
&= \left[\left(c_{i+1+j} - c_{i+2+j}\right)\mu_{i+1+j} - \left(c_i - c_{i+1}\right)\mu_i\right]E\sum_{s=1}^{\sigma+1}\beta^s > \mathbf{0}\,.
\end{aligned}$$

- And, by induction, no policy other than π can be optimal.

■

Proposition 2:

Let i, j be nonnegative integers, and suppose the following condition holds:

— **Condition C3**

$$\left(c_i - c_{i+1}\right)\mu_i \geq \left(c_{i+1+j} - c_{i+2+j}\right)\mu_{i+1+j}$$

When $i+2+j \leq N$ further assume that:

— **Condition C4**

$$Q_{i+1+j} > Q_{i+2+j}$$

Then:

$$Q_i > Q_{i+1+j}$$

Proof: Since it follows the same type of argument used in Proposition 1, it will be omitted.

■

4. OPTIMAL POLICIES FOR THREE QUEUES IN TANDEM

We now apply the results obtained above to the particular scenario of three queues in tandem. In order to ease the notation we define:

$$a_i = \left(c_i - c_{i+1}\right)\mu_i \qquad i = 1,2$$
$$= c_3\mu_3 \qquad i=3$$

Note that $a_i = \Delta_i^1$, so it is the expected savings in the one step ahead cost when the server is assigned to Q_i. A policy which minimizes the one step ahead cost is called " myopic ". In our problem a policy is myopic if: $a_i > a_j \rightarrow Q_i > Q_j$. By virtue of the definition of a_i, we call it a "$c\mu$ *type*" policy. We are interested in investigating under which conditions, if any, a myopic policy is optimal for the posed problem. Furthermore, we will find an optimal policy when the myopic one does not prove to be optimal.

It turns out that the optimal policy depends on the relative values of a_i. For some combinations of the relative values of a_i it also depends on the value of β, and λ. In the following set of claims we cover all possible cases.

Claim 1.
Suppose that:

$$a_1 \geq a_2 \geq a_3$$

Then:

$$Q_1 > Q_2 > Q_3$$

Proof: Follows immediately from Proposition 1.

■

Claim 2.
Suppose that:

$$a_1 \geq a_3 > a_2$$

Then:

$$Q_1 > Q_3 > Q_2$$

Proof:

Proposition 1 --> $Q_3 > Q_2$

Proposition 2 --> $Q_1 > Q_3$

It only rests to show that $Q_1 > Q_3$. Whenever $x_3 > 0$ the precedence order is clearly given by $Q_1 > Q_3 > Q_2$. An interchange argument similar to the one used in Proposition 1 shows that Q_1 has priority over Q_2 even when $x_3 = 0$.

■

Claim 3:
Suppose that:

$$a_2 > a_1 \geq a_3$$

Then:

$$Q_2 > Q_1 > Q_3$$

Proof:

Proposition 1 --> $Q_2 > Q_1$

Proposition 2 --> $Q_2 > Q_3$

Proposition 2 --> $Q_1 > Q_3$

and the claim follows.

■

Claim 4:
Suppose that:

$$a_3 > a_2 > a_1$$

Then:

$$Q_3 > Q_2 > Q_1$$

Proof:

Proposition 1 --> $Q_3 > Q_2$

Proposition 1 --> $Q_2 > Q_1$

Clearly, whenever $x_2 > 0$ the claim holds. An interchange argument similar to the one used in Proposition 1 completes the proof when $x_2 = 0$.

■

Claim 5.
Suppose that:

$$a_2 \geq a_3 > a_1$$

and

$$\delta \geq 0$$

where:

$$\delta = c_3\mu_3 - \mu_1\left(c_1 - c_2\right) + \mu_1\frac{\mu_3}{r}c_3\frac{\beta}{1. - \beta}\left(1. - E\beta^{\sigma}\right) - \mu_1\left(c_2 - c_3\right)E\beta^{\sigma} \tag{4.1}$$

and:

$$E\beta^{\sigma} = \beta\frac{\mu_2}{r}\frac{1.}{1. - \beta\left(1. - \frac{\mu_2}{r}\right)} \tag{4.2}$$

Then:

$$Q_2 > Q_3 > Q_1$$

Proof of the claim:

- The following facts follow from Propositions 1 and 2:

$$Q_2 > Q_1$$

$$Q_2 > Q_3$$

- The situation when $x_2 > 0$ is then clear. All we need to show now is that, whenever $x_2 = 0$, the optimal policy serves Q_3.

  We are going to use induction. As before,the last step in the induction will be proved by contradiction, and the contradiction will be obtained through a more delicate interchange argument. In this case, it is not possible to interchange two decisions without perturbing the others, because serving Q_1 inserts, with probability $\frac{\mu_1}{r}$, a customer in Q_2. Under the hypothesis of the claim, it is optimal to serve Q_2 whenever it is not empty. We incorporate this fact in the following.

  — Let n denote the number of stages to go, and π a policy which is optimal if the claim holds.

  — The claim trivially holds for $n = 1$. It also holds for $n = 2$ (the myopic policy is optimal when there is only one stage to go).

  — Let us assume that π is optimal for some generic $n > 2$.

  — We want to prove that π is then optimal for $n+1$. In fact, we only need to test it when $x_2(n+1) = 0$. So, we assume that both Q_1 and Q_3 have customers and that Q_2 is empty at $n+1$. By way of contradiction, let us assume there exists an optimal policy $\tilde{\pi}$, which serves Q_1 at epoch $n+1$, and which is identical to π thereafter. The act of serving Q_1 at $n+1$ initiates a cycle of Q_2 services of length σ, where σ is the (random) number of service periods it takes to serve a customer in Q_2. Q_3 is eventually served at $n-\sigma$. (Without loss of generality $\sigma \leq n-1$). Also note that $\sigma = 0$ with probability $1. - \frac{\mu_1}{r}$.

  — Consider now the policy $\hat{\pi}$, which interchanges the cycle of decisions started when Q_1 is served, and the decision at $n-\sigma$. So, Q_3 is served at $n+1$, Q_1 at n, and the cycle induced by Q_1 in the interval $\left[n-1, n-\sigma\right]$. Beyond $n-\sigma$ the decisions taken by both policies are identical. The relative expected savings obtained by using policy $\hat{\pi}$ instead of $\tilde{\pi}$ are given by:

$$\begin{aligned}
\Delta\left(\hat{\pi} - \tilde{\pi}\right) &= \left(1. - \frac{\mu_1}{r}\right)\frac{c_3}{r}\beta + \frac{\mu_1}{r}\left\{c_3\frac{\mu_3}{r}E^n\sum_{s=1}^{\sigma+1}\beta^s - \left(c_1 - c_2\right)\beta - \left(c_2 - c_3\right)E^n\beta^{\sigma+1}\right\} \\
&= \frac{\beta}{r}c_3\mu_3\left(1. - \frac{\mu_1}{r}\right) \\
&\quad + \beta\frac{\mu_1}{r}\left\{c_3\frac{\mu_3}{r}\frac{1.}{1.-\beta} - \left(c_1 - c_2\right) - \left[c_3\mu_3\frac{\beta}{1.-\beta} + \left(c_2 - c_3\right)\right]E^n\beta^\sigma\right\} \\
&= \frac{\beta}{r}\left[c_3\mu_3 - \mu_1\left(c_1 - c_2\right)\right] + \frac{\beta}{r}\mu_1\frac{\mu_3}{r}c_3\frac{\beta}{1.-\beta}\left(1. - E^n\beta^\sigma\right) - \frac{\beta}{r}\mu_1\left(c_2 - c_3\right)E^n\beta^\sigma \\
&> \frac{\beta}{r}\delta \geq 0\,.
\end{aligned}$$

where:

$$E^n\beta^\sigma = \sum_{s=1}^{n-1} \beta^s \frac{\mu_2}{r}\left(\frac{\mu_2}{r}\right)^{s-1} = \beta\frac{\mu_2}{r}\frac{1. - \beta\left(1 - \frac{\mu_2}{r}\right)^n}{1. - \beta\left(1. - \frac{\mu_2}{r}\right)} \tag{4.3}$$

So, the policy $\bar{\pi}$ cannot be optimal, and we have thus proved by induction that it is optimal to serve Q_3 when $x_2 = 0$.

■

Claim 6:
Suppose that:

$$a_2 \geq a_3 > a_1$$

and

$$\delta < 0$$

where δ has been defined in (4.3).
Then:

1. The optimal policy for the infinite horizon policy is stationary and follows the non-myopic rule:

$$Q_2 > Q_1 > Q_3$$

2. The finite horizon policy is not stationary.

Proof: Recall the definition of policies $\hat{\pi}$, $\bar{\pi}$ and $\Delta\left(\hat{\pi} - \bar{\pi}\right)$ given in the proof of Claim 5. Also note that $\Delta\left(\hat{\pi} - \bar{\pi}\right)$ converges to $\frac{\beta}{r}\delta$. Therefore, for n sufficiently large, the policy $\bar{\pi}$ outperforms $\hat{\pi}$. This means that it cannot be optimal to serve Q_3 at the first stage (instead of serving Q_1) when $x_2 = 0$. In a similar fashion, the expected savings obtained by postponing the service of Q_3 for m stages and serving Q_1 (Q_2 if not empty) meanwhile converges to $(1 + \beta + \ldots + \beta^{m-1})\left(-\frac{\beta}{r}\delta\right)$ as the length of the planning horizon goes to infinity. Then, the optimal stationary policy is never to serve Q_3 unless both Q_2 and Q_1 are empty. To complete the proof, note that when the planning horizon is 2, the "myopic policy " is optimal, so the optimal policy for the finite horizon is not stationary.

■

Claim 7.
Suppose that:

$$a_3 > a_1 \geq a_2$$

and:

$$\gamma \geq 0$$

where:

$$\gamma = \mu_1\left(c_1 - c_2\right) - \mu_2\left(c_2 - c_3\right) + \frac{\mu_1}{r}\mu_2\left(c_1 - c_2\right)\frac{\beta}{1. - \beta}\left(1. - E\beta^\tau\right) - c_3\mu_2 E\beta^\tau, \tag{4.4}$$

$$E\beta^\tau = \beta\frac{\mu_3}{r}\frac{1.}{1. - \beta\left(1. - \frac{\mu_3}{r}\right)}, \tag{4.5}$$

and:

τ = *the random number of service periods it takes to serve a customer in* Q_3.
Then:

$$Q_3 > Q_1 > Q_2$$

Proof: Proposition 1 and Proposition 2 show that

$Q_3 > Q_2$

$Q_3 > Q_1$

The rest of the proof is similar to the one given in Claim 5. ■

Claim 8.
Suppose that:

$$a_3 > a_1 \geq a_2$$

and:

$$\gamma < 0$$

where γ has been defined in (4.4).

Then:

1. The optimal policy for the infinite horizon problem is stationary and follows the non-myopic rule:

$$Q_3 > Q_2 > Q_1$$

2. The optimal policy for the finite horizon problem is not stationary.

Proof: It uses the same line of argument given in the proof of Claim 6. ■

Proposition 3:

Consider the posed problem of three queues in tandem. Let π^*_α denote a policy which minimizes the α discounted, infinite horizon performance of this system within the class of preemptive policies (i.e. (2.2) with $T = \infty$), derived according to the claims discussed above. Let π^*_{KL} be the optimal stationary policy which minimizes the long run average performance of the same system, within the class of non-preemptive policies, derived according to Klimov's *index rule*[2]. Then,

$$\lim_{\alpha \to 0} \pi^*_\alpha = \pi^*_{KL} \tag{4.6}$$

Proof:

Recall that $\alpha \to 0$ implies $\beta \to 1$. Note that both δ (defined by (4.1)) and γ (defined by (4.4)) have a limit as $\beta \to 1$. Therefore, $\lim_{\alpha \to 0} \pi^*_\alpha$ exists. Straightforward algebra completes the proof. ■

Corollary:

Consider the posed problem. The policy which minimizes the long run average cost within the class of non-preemptive policies is also optimal when preemption is allowed.

Proof:

In view of Proposition 3, all we need to show is that π^*_{KL} mininimizes the long run average cost within the class of preemptive policies.

It is well known[9] that, for any policy π which induces a single ergodic class,
$\lim_{\beta \to 1} (1 - \beta)\, J^\beta_\pi(x) = J^{av}_\pi$

where $J_\pi^\beta(x)$ is the β discounted cost to go when the system starts at x, and J_π^{av} is the long run average cost.

Recall the definition of π_α^*. Note that this preemptive policy depends on α at most through the sign of δ or the sign of γ. Therefore, since both δ and γ have a limit as $\alpha \to 0$, there exists some $\tilde{\alpha}$ such that, for any $\alpha < \tilde{\alpha}$, $\pi_\alpha^* = \pi_{KL}^*$. (Note that π_{KL}^* here, although derived according to Klimov's rule, is implemented as a preemptive policy).

Then, for α small enough (β close enough to 1),

$$J_{\pi_{KL}^*}^\beta \le J_\pi^\beta$$

for any non-anticipative policy π. Therefore, multiplying by $1 - \beta$ and taking limits as $\beta \to 1$,

$$J_{\pi_{KL}^*}^{av} \le J_\pi^{av}$$

■

5. CONCLUSIONS

We have derived an optimal dynamic scheduling policy for three queues in tandem which compete for a single server, for a wide possible combination of the coefficients in the linear cost function.

The optimal control law which assigns the server depends on the state of occupancy of the queues (i.e. empty or not empty), but not on the actual number of customers. For the infinite horizon case it admits a very simple *static priority rule* which depends both on the parameters of system (arrival rate, service rates) and the parameters of the performance measure (cost coefficients and discount factor). Unlike the case of queues in parallel this rule is non-myopic and depends on the arrival rate for some combinations of the above parameters.

REFERENCES

1. G.P. Klimov, "Time-Sharing Service Systems.I", Theory of Prob. and its Applications", vol. 19, # 3, pp. 532-551, 1974.
2. G.P. Klimov, "Time-Sharing Service Systems.II", Theory of Prob. and its Applications", vol. 23, # 2, pp. 314-321, 1978.
3. J.S. Baras, A.J. Dorsey and A.M. Makowski, "Two competing queues with linear costs and geometric service requirements: The μc -rule is often optimal", Adv. Appl. Prob., vol. 17, pp. 186-209, 1985.
4. C. Buyukkoc, P. Varaiya and J. Walrand, "The $c\mu$ Rule Revisited ", Adv. Appl. Prob., vol. 17, pp. 237-238, 1985.
5. L. Kleinrock, "Queueing Systems", vol. II, John Wiley & Sons, 1976.
6. S. Lippman, "Applying a New Device in the Optimization of Exponential Queueing Systems", Opns.Res. 23,687-710 (1975).
7. R. Serfozo, "An equivalence between discrete and continuous time Markov decision processes,", Oper. Res., vol.27,pp. 616-620, 1979
8. B. Hajek, "Optimal Control of Two Interacting Service Stations", IEEE Trans. Automat. Contr.,vol. AC-29, no.6, pp. 491-499, 1984.
9. S. Lippman, "Semi-Markov decision processes with unbounded rewards", Management Sci., vol. 19, pp. 717-731, Mar. 1973.

Geometric Aggregation, Expansion and Partial Observations for Markov Processes

Josep M. Ferrandiz and Aurel A. Lazar
Department of Electrical Engineering
and
Center for Telecommunications Research
Columbia University, New York, NY 10027

ABSTRACT

Geometric aggregations, expansions and partial observations of the state space of stationary Markov processes are considered. Based on a martingale approach, necessary and sufficient conditions for the stochastic process obtained after geometric agrregation to be Markovian are given. It is shown that the conditional equilibrium probabilities of a Markov process coincide with the equilibrium probabilities if and only if the Muntz-Melamed balance equations hold. The stochastic process obtained after geometric aggregation is Markovian if and only if the associated reversed process satisfies the Muntz-Melamed balance equations.

1. Introduction

In many problems involving random phenomena one is led to consider stochastic processes that are a function of a Markov process. It is natural to inquire under what circumstances the resulting process is also Markovian. In this paper, the processes resulting from geometrically aggregating and expanding states of the original Markov process are considered.

Kemeny and Snell [KEME76] gave, in the case of discrete time Markov chains, necessary and sufficient conditions for the process obtained by aggregation of states to be Markovian. These conditions will be referred to as the aggregation conditions. The aggregation conditions allow the definition of the transition rates of the resulting process in a natural way. The aggregation technique is useful in reducing the original state space and decreasing the complexity of the problem of finding the equilibrium probabilities of the original process. Note, however, that the steady state probabilities of the embedded process are in general *not equal* to the equilibrium probabilities of the original Markov process.

A closely related problem to the one described above arises in many practical applications where only partial observations of the Markov process transitions are available. In this case the observed process is embedded in the original process and one can then ask what are necessary and sufficient conditions for the original Markov process to have its steady state probabilities equal to the equilibrium probabilities of the embedded process. This question can be answered using a martingale approach [BREM81], [BREM85]. The observed equilibrium probabilities are *equal* to the original ones if and only if a set of partial balance equations hold. Following Brémaud [BREM81], these equations will be referred to as the Muntz-Melamed balance equations.

The inverse process of state aggregation is called state expansion. Under geometric expansion, the state space is increased and the original Markov process is embedded into another Markov process which will be referred to as the expanded process. In [BEUE86], a state expansion is used to find a recursion for the equilibrium probalities of a class of Quasi-Birth-and-Death processes.

In this paper, the aggregation conditions are derived for continuous time Markov chains using an integral representation of these processes. It is proved that the Muntz-Melamed balance equations are equivalent to the aggregation conditions for the reversed process. The structure and properties of state expansions are investigated. Finally, the expanded process is constructed and it is shown that state space expansions help in understanding the probabilistic behavior of the original process.

The paper is organized as follows. Section 2 contains a new derivation of the aggregation conditions for Markov processes. In Section 3, the Muntz-Melamed balance equations are derived. The equivalence between the Muntz-Melamed equations and the aggregation conditions for the reversed Markov chain is shown in Section 4. Some implications of this equivalence are also presented. Section 5 contains the description of some properties of the state expansion technique. The conclusions of this paper are given in Section 6.

2. Aggregated Markov Processes

Let $(\Omega, \mathcal{F}, \mathrm{P})$ be a probability space and let $(X_t, \mathcal{F}_t^X)_{t\in \mathbf{R}_+}$ be a stationary positive recurrent Markov process in equilibrium, with a countable state space S, a bounded conservative infinitesimal generator $Q = (q_{ij})$, *i.e.*, such that $q_i = \sum_{j\neq i} q_{ij} < \infty$ and equilibrium probability vector $(\pi_i)_{i\in S}$.

In what follows, we shall summarize a construction method for the Markov process $(X_t)_{t\in \mathbf{R}_+}$ which will be needed in the proof. For a more detailled account of this method, see [BREM85] and [BREM81].

Let i and j $(i \neq j)$ be elements of S. On $(\Omega, \mathcal{F}, \mathrm{P})$, let $(\tilde{N}_{ij}(t), \mathcal{F}_t^X)_{t\in \mathbf{R}_+}$ be independent Poisson processes with intensities q_{ij}. Finally, consider processes $V_i(t)$, $i \in S$, with values in $\{0,1\}$, defined by,

$$\sum_{i\in S} V_i(t) = 1$$

and

$$V_i(t) = V_i(0) + \sum_{j\neq i} \int_0^t V_j(s^-)\, d\tilde{N}_{ji}(s) - \int_0^t V_i(s^-) \sum_{j\neq i} d\tilde{N}_{ij}(s). \tag{1}$$

Then, the process $(Y_t)_{t\in\mathbf{R}_+}$ defined by

$$Y_t = i \text{ if and only if } V_i(t) = 1$$

is a Markov process with infinitesimal generator $Q = (q_{ij})$ [BREM85]. Since the processes $(X_t)_{t\in\mathbf{R}_+}$ and $(Y_t)_{t\in\mathbf{R}_+}$ are probabilistically equivalent, $(X_t)_{t\in\mathbf{R}_+}$ will denote the representative of this equivalence class.

From this construction method it also follows that the number of transitions of $(X_t)_{t\in\mathbf{R}_+}$ from state i into state j in $[0,t)$ is given by

$$N_{ij}(t) = \int_0^t Z_i(s^-)\, d\tilde{N}_{ij}(s), \tag{2}$$

where $Z_i(t) = \mathbf{1}(X_t = i)$. Therefore, the number of transitions from state i into E_β in $[0,t)$ is

$$N_{iE_\beta}(t) = \sum_{j\in E_\beta} N_{ij}(t)$$

and the number of transitions of $(X_t)_{t\in\mathbf{R}_+}$ from E_α into E_β in $[0,t)$ is

$$N_{E_\alpha E_\beta}(t) = \sum_{i\in E_\alpha} N_{iE_\beta}(t).$$

Let $\mathcal{E} = (E_\alpha)_{\alpha\in\mathcal{T}}$ be a countable partition of the state space $\mathcal{S}$ and define a stochastic process $(X_t^*)_{t\in\mathbf{R}_+}$ by:

$$X_t^* = \alpha \quad \text{if and only if} \quad X_t \in E_\alpha\,.$$

Intuitively, this corresponds to aggregating all the states of $(X_t)_{t\in\mathbf{R}_+}$ in E_α into a single state denoted by α . Then, the state space of the process $(X_t^*)_{t\in\mathbf{R}_+}$ can be taken to be $\mathcal{T}$, the index set of the partition $\mathcal{E}$.

For clarity, the elements of the state space $\mathcal{S}$ of the original process X_t will be denoted by i, j, k, and those of the state space $\mathcal{T}$ of the process X_t^* by α, β, γ. The probabilistic behavior of $(X_t^*)_{t\in\mathbf{R}_+}$ is fully described by the initial probability distribution, $\mathrm{P}(X_0^* = \alpha_0) = \mathrm{P}(X_0 \in E_{\alpha_0})$, and the conditional probabilities

$$\begin{aligned}\mathrm{P}(X_{t_n}^* = \alpha_n | X_0^* = \alpha_0, X_{t_1}^* = \alpha_1, \ldots, X_{t_{n-1}}^* = \alpha_{n-1}) =& \\ = \mathrm{P}(X_{t_n} \in E_{\alpha_n} | X_0 \in E_{\alpha_0}, \ldots, X_{t_{n-1}} \in E_{\alpha_{n-1}}),&\end{aligned}$$

where $(t_n)_{n\in\mathbf{N}}$ is any increasing sequence of positive real numbers. The definition of an aggregated process can now be introduced.

Definition 1. *The process $(X_t)_{t\in\mathbf{R}_+}$ can be geometrically aggregated with respect to a partition $\mathcal{E}$ if the process $(X_t^*)_{t\in\mathbf{R}_+}$ is Markovian.*

Definition 2. *If the process $(X_t)_{t\in\mathbf{R}_+}$ can be geometrically aggregated, $(X_t^*)_{t\in\mathbf{R}_+}$ is called the aggregated process.*

The number of transitions of $(X^*_t)_{t\in\mathbf{R}_+}$ from state α into state β in $[0,t)$ is denoted by $N^*_{\alpha\beta}(t)$, for α and β $(\alpha \neq \beta)$ in $\mathcal{T}$. Let also $q_{kE_\beta} = \sum_{l\in E_\beta} q_{kl}$ and $Z^*_\alpha(t) = \mathbf{1}(X^*_t = \alpha)$. Two technical lemmas are presented next.

Lemma 1. $N^*_{\alpha\beta}(t) = N_{E_\alpha E_\beta}(t)$.

Proof: The definition of $(X^*_t)_{t\in\mathbf{R}_+}$ implies that

$$\mathbf{1}(X^*_t = \alpha) = \mathbf{1}(X_t \in E_\alpha) = \sum_{i\in E_\alpha} \mathbf{1}(X_t = i).$$

By simple algebraic manipulations it follows,

$$\begin{aligned}
N^*_{\alpha\beta}(t) &= \sum_{0\leq s<t} \mathbf{1}(X^*_{s-} = \alpha, X^*_s = \beta) = \sum_{0\leq s<t} \mathbf{1}(X^*_{s-} = \alpha)\mathbf{1}(X^*_s = \beta)\\
&= \sum_{0\leq s<t}\sum_{i\in E_\alpha} \mathbf{1}(X_{s-} = i)\sum_{j\in E_\beta}\mathbf{1}(X_s = j) = \sum_{(i,j)\in E_\alpha\times E_\beta}\ \sum_{0\leq s<t} \mathbf{1}(X_{s-} = i, X_s = j)\\
&= \sum_{(i,j)\in E_\alpha\times E_\beta} N_{ij}(t) = N_{E_\alpha E_\beta}(t).
\end{aligned}$$

■

Lemma 2.

$$Z^*_\beta(t) = Z^*_\beta(0) + \sum_{\alpha\neq\beta}\int_0^t \sum_{j\in E_\alpha} Z_j(s^-)\,d\tilde{N}_{jE_\beta}(s) - \sum_{\alpha\neq\beta}\int_0^t\sum_{k\in E_\beta} Z_k(s^-)d\tilde{N}_{kE_\alpha}(s),$$

for all β in $\mathcal{T}$.

Proof: Since $Z^*_\beta(t) = \sum_{k\in E_\beta} Z_k(t)$, we have

$$\begin{aligned}
Z^*_\beta(t) &= \sum_{k\in E_\beta}\left(Z_k(0) + \sum_{j\neq k}\int_0^t Z_j(s^-)\,d\tilde{N}_{jk}(s) - \int_0^t Z_k(s^-)\sum_{j\neq k} d\tilde{N}_{kj}(s)\right)\\
&= Z^*_\beta(0) + \sum_{k\in E_\beta}\sum_{\alpha\in\mathcal{T}}\sum_{\substack{j\in E_\alpha\\ j\neq k}}\int_0^t Z_j(s^-)\,d\tilde{N}_{jk}(s) - \sum_{k\in E_\beta}\sum_{\alpha\in\mathcal{T}}\sum_{\substack{j\in E_\alpha\\ j\neq k}}\int_0^t Z_k(s^-)\,d\tilde{N}_{kj}(s)\\
&= Z^*_\beta(0) + \sum_{\alpha\neq\beta}\int_0^t\sum_{j\in E_\alpha} Z_j(s^-)\,d\tilde{N}_{jE_\beta}(s) + \sum_{\substack{j,k\in E_\beta\\ j\neq k}}\int_0^t Z_j(s^-)\,d\tilde{N}_{jk}(s)\\
&\quad - \sum_{\alpha\neq\beta}\int_0^t\sum_{k\in E_\beta} Z_k(s^-)d\tilde{N}_{kE_\alpha}(s) - \sum_{\substack{j,k\in E_\beta\\ j\neq k}}\int_0^t Z_k(s^-)\,d\tilde{N}_{kj}(s),
\end{aligned}$$

and the Lemma is proved by cancelling the equal terms in the last equality. ■

The aggregation conditions for continuous time Markov processes are presented next.

Theorem 1. *The Markov process* $(X_t)_{t\in\mathbf{R}_+}$ *can be geometrically aggregated with respect to a partition* $\mathcal{E}$ *if and only if* q_{kE_β} *is independent of* k, *for any* α *and* β *in* $\mathcal{T}$, $\alpha \neq \beta$, *and all* k *in* E_α.

Proof: ($\Rightarrow$) Assume first that $(X_t)_{t\in\mathbf{R}_+}$ can be geometrically aggregated. Therefore, $(X_t^*)_{t\in\mathbf{R}_+}$ is a Markov process with an infinitesimal generator $Q^* = (q^*_{\alpha\beta})$ and

$$N^*_{\alpha\beta}(t) = \int_0^t Z^*_\alpha(u^-)\, d\tilde{N}^*_{\alpha\beta}(u), \tag{3}$$

where $(\tilde{N}^*_{\alpha\beta}(t), \mathcal{F}_t^{X^*})_{t\in\mathbf{R}_+}$ is a Poisson process with intensity $q^*_{\alpha\beta}$. Equation (3) implies that for any positive predictable process $(H_t, \mathcal{F}_t^{X^*})_{t\in\mathbf{R}_+}$,

$$\mathbf{E}\int_{\mathbf{R}_+} H_u Z^*_\alpha(u^-) q^*_{\alpha\beta}\, du = \mathbf{E}\int_{\mathbf{R}_+} H_u\, dN^*_{\alpha\beta}(u).$$

From Lemma 1, $N^*_{\alpha\beta}(t) = \sum_{i\in E_\alpha} N_{iE\beta}(t)$ hence,

$$\mathbf{E}\int_{\mathbf{R}_+} H_u\, dN^*_{\alpha\beta}(u) = \sum_{i\in E_\alpha} \mathbf{E}\int_{\mathbf{R}_+} H_u\, dN_{iE_\beta}(u),$$

and

$$\sum_{i\in E_\alpha} \mathbf{E}\int_{\mathbf{R}_+} H_u\, dN_{iE_\beta}(u) = \mathbf{E}\int_{\mathbf{R}_+} H_u \sum_{i\in E_\alpha} Z_i(u^-) q_{iE_\beta}\, du.$$

Therefore,

$$\mathbf{E}\int_{\mathbf{R}_+} H_u Z^*_\alpha(u^-) q^*_{\alpha\beta}\, du = \mathbf{E}\int_{\mathbf{R}_+} H_u \sum_{i\in E_\alpha} Z_i(u^-) q_{iE_\beta}\, du, \tag{4}$$

for all positive predictable processes $(H_t, \mathcal{F}_t^{X^*})_{t\in\mathbf{R}_+}$.

If for every k in E_α, we set $H_t = L_t Z_k(t^-)$, where $(L_t, \mathcal{F}_t^{X^*})_{t\in\mathbf{R}_+}$ is positive and predictable, equation (4) becomes

$$\mathbf{E}\int_{\mathbf{R}_+} L_u Z_k(u^-) Z^*_\alpha(u^-) q^*_{\alpha\beta}\, du = \mathbf{E}\int_{\mathbf{R}_+} L_u Z_k(u^-) \sum_{i\in E_\alpha} Z_i(u^-) q_{iE_\beta}\, du. \tag{5}$$

Since for k in E_α, $Z_k(t) Z^*_\alpha(t) = Z_k(t)$ and $Z_k(t)\sum_{i\in E_\alpha} Z_i(t) q_{iE_\beta} = Z_k(t) q_{kE_\beta}$ it follows from equation (5) that

$$\mathbf{E}\int_{\mathbf{R}_+} L_u Z_k(u^-) q^*_{\alpha\beta}\, du = \mathbf{E}\int_{\mathbf{R}_+} L_u Z_k(u^-) q_{kE_\beta}\, du,$$

for any positive predictable process $(L_t, \mathcal{F}_t^{X^*})_{t\in\mathbf{R}_+}$. Therefore, $q^*_{\alpha\beta} = q_{iE_\beta}$ for any i in E_α and for all α, β in $\mathcal{T}$, $\alpha \neq \beta$.

($\Leftarrow$) Conversely, we have to show that $(X_t^*, \mathcal{F}_t^{X^*})_{t\in\mathbf{R}_+}$ is a Markov process. Since the aggregation conditions hold, let $q^*_{\alpha\beta}$ be the common value of the q_{jE_β} for j in E_α. This common value is the intensity of the family of independent Poisson processes $(\tilde{N}_{jE_\beta}(t), \mathcal{F}_t^X)_{t\in\mathbf{R}_+}$, with j in E_α. Then, for any positive predictable process $(H_t, \mathcal{F}_t^X)_{t\in\mathbf{R}_+}$ we have,

$$\begin{aligned}\mathbf{E}\int_{\mathbf{R}_+} H_s \sum_{j\in E_\alpha} Z_j(s^-)\, d\tilde{N}_{jE_\beta}(s) &= \mathbf{E}\int_{\mathbf{R}_+} H_s \sum_{j\in E_\alpha} Z_j(s^-) q^*_{\alpha\beta}\, ds \\ &= \mathbf{E}\int_{\mathbf{R}_+} H_s Z^*_\alpha(s^-) q^*_{\alpha\beta}\, ds \\ &= \mathbf{E}\int_{\mathbf{R}_+} H_s Z^*_\alpha(s^-)\, dN_{kE_\beta}(s),\end{aligned}$$

for any k in E_α. Therefore, by the projection theorem [BREM81],

$$\sum_{j\in E_\alpha} Z_j(s^-)\, d\tilde{N}_{jE_\beta}(s) = Z^*_\alpha(s^-)\, dN_{kE_\beta}(s),$$

for all k in E_α, (P-a.s.). Similarly, one can show that

$$\sum_{k\in E_\beta} Z_k(s^-)\, d\tilde{N}_{kE_\alpha}(s) = Z^*_\beta(t^-)\, d\tilde{N}_{lE_\alpha}(s),$$

for all l in E_β, (P-a.s.). Thus, from Lemma 2,

$$Z^*_\beta(t) = Z^*_\beta(0) + \sum_{\alpha\neq\beta}\int_0^t Z^*_\alpha(s^-)\, d\tilde{N}_{kE_\beta}(s) - \sum_{\alpha\neq\beta}\int_0^t Z^*_\beta(s^-)\, d\tilde{N}_{lE_\alpha}(s), \tag{6}$$

for all l in E_β and all k in E_α, which is formally identical to equation (1). This shows that $(X^*_t)_{t\in\mathbf{R}_+}$ is a Markov process with infinitesimal generator $Q^* = (q^*_{\alpha\beta})$. ∎

From the previous proof it also follows,

Corollary 1. *The infinitesimal generator of the aggregated process is given by:*

$$q^*_{\alpha\beta} = \sum_{l\in E_\beta} q_{kl},$$

for any k in E_α.

Corollary 2. *The equilibrium probabilities of the aggregated process are given by:*

$$\pi^*_\alpha = \sum_{k\in E_\alpha} \pi_k.$$

Proof: The equilibrium probabilities of $(X_t)_{t\in\mathbf{R}_+}$ satisfy:

$$\sum_{k\in\mathcal{S}} \pi_k q_{kl} = 0 \quad ,$$

for all $l\in\mathcal{S}$. Adding over $l\in E_\beta$ and after some algebraic manipulations,

$$\sum_{l\in E_\beta}\sum_{k\in\mathcal{S}} \pi_k q_{kl} = \sum_{k\in\mathcal{S}} \pi_k q_{kE_\beta} = \sum_{\alpha\in\mathcal{T}}\Big(\sum_{k\in E_\alpha} \pi_k q_{kE_\beta}\Big) = \sum_{\alpha\in\mathcal{T}} \pi^*_\alpha q^*_{\alpha\beta} = 0 \quad ,$$

for all $\beta\in\mathcal{T}$. The last equality follows from the aggregation conditions. ∎

3. Partial Observations

In the sequel we assume that there is a partition $\mathcal{E} = (E_\alpha)_{\alpha \in \mathcal{T}}$ of the state space $\mathcal{S}$ such that for every coset E_β, all the observed transitions of the process $(X_t)_{t \in \mathbf{R}_+}$ come from a *unique* set E_α of $\mathcal{E}$ $(\alpha \neq \beta)$. One can then ask under which conditions the equilibrium probabilities of the observed process are equal to the equilibrium probabilities of the original process.

Remark. This partial observations scheme is a special case of the one presented in [BREM81] and used in the derivation of the Muntz-Melamed balance equations.

Let $N_{ij}(t)$ be the point process that counts the number of transitions of $(X_t)_{t \in \mathbf{R}_+}$ from state i into state j in $(0, t]$, *i.e.*, $N_{ij}(t) = \sum_{0 < s \leq t} \mathbf{1}(X_{s^-} = i, X_s = j)$ and $N^*_\beta(t)$ be the point process that counts all the transitions into E_β, *i.e.*,

$$N^*_\beta(t) = \sum_{(i,j) \in \mathcal{S} \times E_\beta} N_{ij}(t).$$

Since all transitions to E_β come from a unique E_α, we will also denote $N^*_\beta(t)$ by $N^*_{\alpha\beta}(t)$ and

$$N^*_\beta(t) = N^*_{\alpha\beta}(t) = \sum_{(i,j) \in E_\alpha \times E_\beta} N_{ij}(t).$$

Let $\mathcal{F}_t^{N^*} = \bigvee_{\alpha \in \mathcal{T}} \mathcal{F}_t^{N^*_\alpha}$, where $(\mathcal{F}_t^{N^*_\alpha})_{t \in \mathbf{R}_+}$ is a history of $N^*_\alpha(t)$ and let $\pi^*_i = \mathrm{P}(X_t = i | \mathcal{F}_t^{N^*})$ be the probability of being in state i, i in E_β. Under this partial observations scheme, the total probability flow into E_β is given by

$$\sum_{(k,l) \in E_\alpha \times E_\beta} \pi_k q_{kl}.$$

Similarly, if i is in E_β, the total probability flow into i is equal to

$$\sum_{k \in E_\alpha} \pi_k q_{ki}.$$

Therefore, when only transitions from sets E_α into sets E_β are observed, the probability of being in state i is

$$\pi^*_i = \frac{\sum_{k \in E_\alpha} \pi_k q_{ki}}{\sum_{(k,l) \in E_\alpha \times E_\beta} \pi_k q_{kl}}.$$

Thus, [BREM81] and [BREM85]

Theorem 2. *Let $(X_t, \mathcal{F}_t^X)_{t \in \mathbf{R}_+}$ be a Markov process in equilibrium, then*

$$\mathrm{P}(X_t = i | \mathcal{F}_t^{N^*}) = \pi_i \quad \textit{if and only if} \quad \pi_i = \frac{\sum_{k \in E_\alpha} \pi_k q_{ki}}{\sum_{(k,l) \in E_\alpha \times E_\beta} \pi_k q_{kl}}, \tag{7}$$

for all $i \in E_\beta$ and all α and β in $\mathcal{T}$ $(\alpha \neq \beta)$.

The right hand side of equation (7) is a form of partial balance. Following Brémaud [BREM85], the set of equations

$$\pi_i = \frac{\sum_{k \in E_\alpha} \pi_k q_{ki}}{\sum_{(k,l) \in E_\alpha \times E_\beta} \pi_k q_{kl}}, \tag{8}$$

for all $i \in E_\beta$ and all α, β in $\mathcal{T}$ $(\alpha \neq \beta)$ are called the Muntz-Melamed balance equations.

4. Partial Observations and Aggregation

Theorem 3. *Let $(X_t, \mathcal{F}_t^X)_{t\in\mathbf{R}_+}$ be a Markov process in equilibrium, $(\overset{r}{X}_t, \overset{r}{\mathcal{F}_t}^{X})_{t\in\mathbf{R}_+}$ its associated reversed process and $\mathcal{E}$ a partition of the state space $\mathcal{S}$. The Muntz-Melamed balance equations (8) hold for a partition $\mathcal{E}$ if and only if the reversed process $(\overset{r}{X}_t, \overset{r}{\mathcal{F}_t}^{X})_{t\in\mathbf{R}_+}$ can be geometrically aggregated with respect to $\mathcal{E}$.*

Proof: Take $E_\alpha, E_\beta \in \mathcal{E}$ and $i \in E_\beta$ such that the Muntz-Melamed balance equations hold. After some algebraic manipulations we obtain,

$$\sum_{k\in E_\alpha} \pi_k \sum_{l\in E_\beta} q_{kl} = \sum_{k\in E_\alpha} \frac{\pi_k q_{ki}}{\pi_i}, \tag{9}$$

for all $i \in E_\beta$. It is well known that the infinitesimal generator of the reversed process $(\overset{r}{X}_t)_{t\in\mathbf{R}_+}$, $\overset{r}{Q}= (\overset{r}{q}_{ij})_{i,j\in\mathcal{S}}$, is given by

$$\overset{r}{q}_{ij}= \frac{\pi_j q_{ji}}{\pi_i}.$$

Thus, equation (9) can be rewritten as

$$\sum_{k\in E_\alpha} \pi_k \sum_{l\in E_\beta} q_{kl} = \sum_{k\in E_\alpha} \overset{r}{q}_{ik}, \tag{10}$$

for all i in E_β. Following the notation introduced in Section 2, equation (10) becomes

$$\sum_{k\in E_\alpha} \pi_k q_{kE_\beta} = \overset{r}{q}_{iE_\alpha},$$

for all i in E_β. Thus, $\overset{r}{q}_{iE_\alpha}$ is independent of i and from Theorem 1, it follows that the elements of $\overset{r}{Q}= (\overset{r}{q}_{ij})_{i,j\in\mathcal{S}}$ satisfy the aggregation conditions. ∎

Theorems 1 and 3 imply that the Muntz-Melamed balance equations hold if and only if the reversed process is Markovian. Since the Markovian property is conserved after time reversal, it follows that the Muntz-Melamed balance equations hold if and only if the observed process is Markovian.

Theorem 3 shows that the stochastic process resulting from a partial observations scheme such as those considered in the previous section will be Markovian (provided that the original process was Markovian).

Theorem 3 also shows that the process $(\overset{r}{X}_t)_{t\in\mathbf{R}_+}$ can be geometrically aggregated with respect to a partition if and only if the observed traffic is independent of $(X_t)_{t\in\mathbf{R}_+}$, *i.e.*, if and only if $\mathcal{F}_t^{N^*}$ and $\sigma(X_t)$ are independent.

Example: Finite Buffer M/G/1 Queue.

Consider an M/G/1 queue where the service rate is modulated by a Birth-and-Death process, *i.e.*, with a service rate given by $\mu_t = \mu X_t$ where μ is a positive real number and $(X_t)_{t\in\mathbf{R}_+}$ is a Birth-and-Death process taking values in $\{0,1,\ldots,M\}$. Furthermore, $(X_t)_{t\in\mathbf{R}_+}$ is independent of the arrival process of the queueing system. Let Q_t be the number of data packets in the queue, $0 \leq Q_t \leq N$, at time t. By definition, $(X_t, \mathcal{F}_t^X)_{t\in\mathbf{R}_+}$ is a Markov process and with this construct, $((X_t, Q_t), \mathcal{F}_t^X \bigvee \mathcal{F}_t^Q)_{t\in\mathbf{R}_+}$ is also a Markov process with state space $\{0,1,\ldots,M\} \times \{0,1,\ldots,N\}$ and infinitessimal generator $(q_{(i,j),(k,l)})$, $i,k \in \{0,\ldots,M\}$, $j,l \in \{0,\ldots,N\}$. In [VAKI86], Vakil and Lazar used this queueing system to model a Time Division Multiplexing frame. One of the difficulties of the analysis has been that $(Q_t, \mathcal{F}_t^Q)_{t\in\mathbf{R}_+}$ is *not* Markovian.

In order to prove this last statement, we only need to show that the aggregation conditions do not hold. This follows immediatly since the aggregation conditions corresponding to aggregating the states $(0,j),\ldots,(M,N)$,

$$\sum_{k=0}^{M} q_{(i,j),(k,j-1)} = \mu i,$$

depend on i.

5. State Space Expansion

In this section, a Markov process $(X'_t, \mathcal{F}_t^{X'})_{t\in\mathbf{R}_+}$, with state space S' ($S \subset S'$), such that $(X_t, \mathcal{F}_t^X)_{t\in\mathbf{R}_+}$ is its aggregated process, will be constructed. Therefore, under a state expansion, the original Markov process $(X_t, \mathcal{F}_t^X)_{t\in\mathbf{R}_+}$ will be embedded into the Markov process $(X'_t, \mathcal{F}_t^{X'})_{t\in\mathbf{R}_+}$.

For any superset S' of S, there exists a partition $\mathcal{E}'$ of S' with index set S. In order to define the expanded process $(X'_t, \mathcal{F}_t^{X'})_{t\in\mathbf{R}_+}$, we derive the form of its infinitesimal generator.

Let $Q' = (q'_{ij})_{i,j\in S'}$ be the infinitesimal generator of $(X'_t, \mathcal{F}_t^{X'})_{t\in\mathbf{R}_+}$ and assume that every coset E_i is geometrically aggregated into a state which we denote by i, (i in S). Since $(X_t, \mathcal{F}_t^X)_{t\in\mathbf{R}_+}$ is a Markov process, the aggregation conditions must hold for the elements of Q', *i.e.*, $\sum_{l\in E_j} q'_{kl}$ must be independent of k for all $k \in E_i$, and equal to q_{ij} (Corollary 1). Therefore, the total rate from any of the states in E_i into E_j must be equal. There are, however, degrees of freedom on how to direct these rates into states of the coset E_j and on how to choose the transitions within a coset. When this freedom is used properly, the use of the expanded chain may modify the original problem in such a way that the evaluation of the equilibrium probabilities is simplified [BEUE86].

The additional understanding that the expanded process may bring about the original process depens greatly on how the expansion is done. For example, assume that, in the original process, state j can only be reached from any of the states $i_1,\ldots,i_n$. Expanding state j into states $j_1,\ldots,j_n$ such that state j_k can only be reached by i_k allows one to obtain a state transition diagram that is characterized by unique transitions into any state [BEUE86]. Other state expansion procedures [FERR86] may eventually allow to eliminate the boundary effects of finite state transition diagrams of

Markov processes. From Corollary 2 we conclude that if the steady state probabilities of $(X'_t, \mathcal{F}_t^{X'})_{t\in\mathbb{R}_+}$ are $(\pi'_i)_{i\in S'}$, the equilibrium probabilities $(\pi_i)_{i\in S}$ of $(X_t, \mathcal{F}_t^{X})_{t\in\mathbb{R}_+}$ are given by

$$\pi_i = \sum_{k\in E_i} \pi'_k,$$

for all $i \in S$.

6. Conclusions

This paper shows the equivalence between a class of partial observations and geometric aggregation for Markov processes. The result suggests a general technique for the study of embedded Markov chains by considering a class of transformations as presented here. When properly chosen, these transformations reduce the complexity of the problem or show recursive structures that simplify the computation of quantities of interest.

Acknowledgement

The work presented here was supported in part by the National Science Foundation under Grant # CDR-84-21420 and in part by the Office of Naval Research under Contract # ONR-N00010-85-K0371.

References

[BEUE86] Beuerman, S. and Coyle, E., "Closed form Solution to the Stationary Probability Vector of a Quasi-Birth-and-Death Process with a Guard State", *Proceedings of the Twentieth Conference on Information Sciences and Systems*, Princeton University, Princeton, NJ, March 1986, pp. 460-464.

[BREM81] Brémaud, P., *Point Processes and Queues*, Springer-Verlag, New York, 1981.

[BREM85] Brémaud, P., *Théorie des Files d'Attente et de leurs Réseaux*, preprint, 1985.

[FERR86] Ferrandiz, J.M. and Lazar, A.A., A Topological Generation of Markov Chain Lattices, *submitted for publication.* [KEME76] Kemeny, J. and Snell, J.L., *Finite Markov Chains*, Springer-Verlag, New York, 1976.

[VAKI86] Vakil, F. and Lazar, A.A., "Flow Control Protocols for Integrated Networks with Partially Observed Voice Traffic", *IEEE Transactions on Automatic Control,* to appear.

A NONROBUSTNESS RESULT ON THE RATE OF CONVERGENCE FOR MARTINGALES

JOHN M. MORRISON
Dept. of Mathematical Sciences
University of Delaware
Newark, Delaware 19716

GARY L. WISE
Dept. of Electrical and Computer Engineering and Dept. of Mathematics
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

In this paper we explore the convergence rate of the martingale convergence theorem and we show that this rate is not robust as a function of the random variable we are attempting to estimate with a sequential scheme. We then extend these results to a class of operators which generalize conditional expectation.

INTRODUCTION

In many sequential schemes concerned with signal processing, estimation theory, and adaptive procedures, we are confronted with the concern of the improvement in performance as more data is obtained and with the establishment of limiting performance as the amount of data increases. In most practical situations we are given a fidelity criterion with which to measure performance. A key tool in the establishment of performance limits in many procedures is the martingale convergence theorem. For example, in estimating a second order random variable Y, we might be interested in the convergence of $E(Y|X_1, X_2, \dots, X_n)$ as $n \to \infty$, where $X_1, X_2, \dots, X_n$ represent our data, a set of n random variables taking values in a separable metric space. With the help of the martingale convergence theorem, we are able to establish convergence of this sequence of estimators in both the mean square sense and the almost sure sense. For even more general examples of this genre, we refer to [1, 2, 3].

In any practical situation, we are confronted with the difficulty of being able to observe only finite sets of data, which are in turn often distorted. A concern which is of paramount importance is getting control of the rate of convergence of the scheme to the asymptotic limit. In this regard, the martingale convergence theorem is the mechanism used to establish the desired convergence; thus we are led to investigate the rate of convergence in the martingale convergence theorem as the means of establishing the speed of convergence of the scheme of interest. In most practical situations we would also be interested in the robustness properties of the desired scheme, and this in turn might often be reflected in the robustness properties of the martingale convergence theorem. In this paper we will investigate the convergence rate and a robustness property of the martingale convergence theorem. We will establish a nonrobustness result for the martingale convergence theorem, and we will also establish the nonuniform convergence of the martingale convergence theorem and discuss consequences of it. Finally, we will extend these results to a class of operators which generalize conditional expectation and which we have shown [1, 2, 3] possess a martingale-like convergence property.

In addition to the foregoing practical considerations, we note that the martingale convergence theorem is one of the key theorems in probability theory. Thus from the perspective of the mathematician, this convergence question is of great theoretical interest.

DEVELOPMENT

Now we will set the mathematical framework for our discussion. Let $(\Omega, \mathfrak{I}, P)$ be a probability space, and let $\Phi:[0, \infty) \to [0, \infty)$ be a convex, increasing function satisfying $\Phi(0) = 0$ and satisfying the doubling condition given by: there exist C and M positive so that $x \geq M$ implies that $\Phi(2x) \leq C\, \Phi(x)$. Let $L^{\Phi}(\Omega, \mathfrak{I}, P)$ be the Orlicz space equipped with the Luxemburg norm corresponding to Φ. For the essential details on these spaces we refer to [2] and [4]. We note that $L^{\Phi}(\Omega, \mathfrak{I}, P)$ is a Banach space. We say that the probability space $(\Omega, \mathfrak{I}, P)$ is separable if for any $p \geq 0$, $L^p(\Omega, \mathfrak{I}, P)$ is a separable metric space. We recall the following result [5, p. 376] pertaining to separability of probability spaces.

Theorem 1: Let $(\Omega, \mathfrak{I}, P)$ be a probability space. Then the following are equivalent:

(i.) There exists $p \geq 0$ so that $L^p(\Omega, \mathfrak{I}, P)$ is a separable metric space.

(ii.) For any $p \geq 0$, $L^p(\Omega, \mathfrak{I}, P)$ is a separable metric space.

(iii.) If we identify sets equal almost everywhere, then $(\mathfrak{I}, \rho)$ is a separable metric space, where $\rho(A, B) = P(A \Delta B)$.

In this note we show that the convergence guaranteed by the martingale convergence theorem is not uniform on any nonvoid open L^{Φ}-ball on any separable probability space whose L^{Φ} space is infinite dimensional. This spectaculary demonstates the nonrobustness of the speed of convergence associated with the martingale convergence theorem.

Consider a sequence $\{\Sigma_n\}_{n \in N}$ of σ-subalgebras of $\mathfrak{I}$ and a σ-subalgebra Σ of $\mathfrak{I}$. We use the notation $\Sigma_n \uparrow \Sigma$ to mean that Σ_n is a subset of Σ_{n+1} for all $n \in N$ and that Σ is the smallest σ-algebra on Ω including Σ_n for all $n \in N$.

We now briefly adumbrate the development on projection schemes as found in [6, section 21].

Definition: Let B be a Banach space and $\{P_n\}_{n \in N}$ be a sequence of continuous linear finite rank projections on B. The sequence $\{P_n\}_{n \in N}$ is called a projection scheme if for each $x \in B$, $P_n(x) \to x$; equivalently, we may require that $\{P_n\}_{n \in N}$ converge to the identity in the strong operator topology.

This is exactly the scenario that one encounters in the martingale convergence theorem. Let $(\Omega, \mathfrak{I}, P)$ be a probability space. Let $\{\Sigma_n\}_{n \in N}$ and Σ be σ-subalgebras of $\mathfrak{I}$ such that $\Sigma_n \uparrow \Sigma$. In practice, the σ-subalgebras Σ_n are generated by data stored on digital computers and are hence finite. In any man-made system and in a large family of natural systems observed in nature there are only a finite number of components which interact; for example, there are only a finite number of atoms in the milky way. In this case, the operators $P_n = E(\cdot\,|\Sigma_n)$, $n \in N$, form a projection sheme on $L^{\Phi}(\Omega, \mathfrak{I}, P)$. The following theorem [6, p. 258] paints a very dismal picture for the robustness in the random variable we are trying to estimate of the rate of convergence guaranteed by the martingale convergence theorem.

Theorem 2: Let B be a Banach space with a projection scheme $\{P_n\}_{n\in N}$. Then for K a nonvoid subset of B, $\sup\{\, \| P_n(x) - x \| : x \in K\} \to 0$ as $n \to \infty$ if and only if K is precompact in B.

Since any locally compact Banach space is finite dimensional [7, p. 37], we immediately have the following.

Corollary 3: Let B be an infinite dimensional Banach space. Then if $\{P_n\}_{n\in N}$ is a projection scheme on B, for any subset K of B having nonvoid interior we have

$$\lim_{n\to\infty} \sup\{\, \| P_n(x) - x \| : x \in K\,\} > 0.$$

In particular, for any $\delta > 0$,

$$\lim_{n\to\infty} \sup\{\, \| P_n(x) - x \| : \| x \| < \delta\} > 0.$$

Suppose $\{\Sigma_n\}_{n\in N}$ is a sequence of finite σ-subalgebras of $\mathfrak{I}$ and Σ is a σ-subalgebra of $\mathfrak{I}$ with $\Sigma_n \uparrow \Sigma$. Consulting the martingale convergence theorems developed in [8], it is easy to see that $\{E(\cdot\,|\,\Sigma_n)\}_{n\in N}$ is a projection scheme on the Banach space $L^{\Phi}(\Omega, \mathfrak{I}, P)$. The following lemma shows that any separable probability space possesses such a finite collection of σ-subalgebras.

Lemma 4: Let $(\Omega, \mathfrak{I}, P)$ be a separable probability space. Then there exists a sequence $\{\Sigma_n\}_{n\in N}$ of finite σ-subalgebras of $\mathfrak{I}$ such that $\Sigma_n \uparrow \mathfrak{I}$.

Proof: Theorem 1 implies that there exists a sequence of events $\{A_n\}_{n\in N}$ such that $\sigma(A_n\!: n\in N)$ and $\mathfrak{I}$ differ only by null sets. Now for each $n \in N$ let $\Sigma_n = \sigma(A_i\!: 1 \le i \le n)$, and the result follows immediately.

QED

Consider any $\Phi\!:[0, \infty) \to [0, \infty)$ that is a convex, increasing function satisfying $\Phi(0) = 0$ and satisfying the doubling condition, and note that

$L^1(\Omega, \mathfrak{I}, P) \supset L^{\Phi}(\Omega, \mathfrak{I}, P) \supset L^{\infty}(\Omega, \mathfrak{I}, P)$. Futher, note that $L^{\infty}(\Omega, \mathfrak{I}, P)$ is finite dimensional if and only if Ω can be written as a finite union of P-atoms, and Ω can be written as a finite union of P-atoms if and only if $L^1(\Omega, \mathfrak{I}, P)$ is finite dimensional. This line of reasoning immediately leads to the following result.

Theorem 5: Let $(\Omega, \mathfrak{I}, P)$ be a probability space. Then the following are equivalent:

(i.) For all $\Phi:[0, \infty) \to [0, \infty)$ that are convex, increasing functions satisfying $\Phi(0) = 0$ and satisfying the doubling condition, $L^{\Phi}(\Omega, \mathfrak{I}, P)$ is infinite dimensional.

(ii.) The Banach space $L^1(\Omega, \mathfrak{I}, P)$ is infinite dimensional.

(iii.) There exists $\Phi:[0, \infty) \to [0, \infty)$ that is a convex, increasing function satisfying $\Phi(0) = 0$ and satisfying the doubling condition, so that $L^{\Phi}(\Omega, \mathfrak{I}, P)$ is infinite dimensional.

(iv.) The set Ω cannot be written as a finite union of P-atoms.

We now exploit this in the following theorem, which is a direct consequence of the foregoing discussion.

Theorem 6: Let $(\Omega, \mathfrak{I}, P)$ be a separable probability space such that Ω cannot be written as a finite union of P-atoms. Then there exists an increasing sequence $\{\Sigma_n\}_{n\in N}$ of σ-subalgebras of $\mathfrak{I}$ so that the L^{Φ} convergence of $\{E(\cdot\,|\Sigma_n)\}_{n\in N}$ is nonuniform on any nonvoid open L^{Φ} ball.

In Theorem 6 we may localize the problem and avoid the need for separability of the probability space. Simply observe that if $(\Omega, \mathfrak{I}, P)$ is not separable, it admits the definition of a nonsimple real valued random variable X; we may then apply the theorem to the separable probability space $(\Omega, \sigma(X), P)$.

In light of this situation, the convergence of the estimators $\hat{E}_{\Phi}(\cdot\,|\Sigma_n)$ that are constructed in [1] and [2] appear to suffer the same shortcomings as their linear cousin, i.e. conditional expectation, discussed here. In fact, in the case $\Phi(x) = x^2/2$ for $x \geq 0$, the linear and nonlinear cases agree. Thus, in general, we do not have a uniform mode of convergence of $\hat{E}_{\Phi}(\cdot\,|\Sigma_n)$ estimators on the L^{Φ} spaces.

Finally, notice that the convergence rate of the martingale convergence theorem, as well as the convergence rate of the estimators $\hat{E}_{\Phi}(\cdot\,|\Sigma_n)$ considered in [1] and [2], can be arbitrarily slow. That is, if the convergence rate were not arbitrarily slow, then the convergence would be uniform. We therefore see that the convergence rate of a martingale sequence associated with an L^{Φ} random variable is an ill-posed property of that random variable with respect to its L^{Φ} norm. This should be a source of concern to researchers working with such martingale based schemes.

The foregoing development tells us that the problem behind controlling the rate of convergence of martingale-based schemes is far more subtle than any information conveyed by the norm of the random variable we trying to estimate. The discovery and quantification of this property constitutes a significant research area awaiting further development.

ACKNOWLEDGEMENT

This research was supported by the Air Force Office of Scientific Research under Grant AFOSR-86-0026.

REFERENCES

[1] J. M. Morrison and G. L. Wise, "A Well-Posedness Property of a Class of Variational Problems and Its Application to Nonlinear Estimation," *Proceedings of the Twenty-Third Annual Allerton Conference on Communication, Control, and Computing*, Monticello, Illinois, October 2-4, 1985, pp. 28-34.

[2] J. M. Morrison and G. L. Wise, "An Asymptotic Property of Nonlinear Estimators Arising as Solutions to a Certain Class of Convex Programming Problems," *Systems and Control Letters*, Vol. 7, pp. 225-232, June 1986.

[3] J. M. Morrison and G. L. Wise, "A Stability Property of Conditional Expectations," *Proceedings of the Nineteenth Annual Conference on Information Sciences and Systems*, Baltimore, Maryland, March 27-29, 1985, pp. 226-229.

[4] M. Krosnoselskii and Y. Rutickii, *Convex Functions and Orlicz Spaces*, Groningen: Noordhoff, 1961.

[5] A. Brown and C. Pearcy, *Introduction to Operator Theory I Elements of Functional Analysis*, New York: Springer Verlag, 1977.

[6] K. Deimling, *Nonlinear Functional Analysis*, New York: Springer Verlag, 1985.

[7] S. Lang, *Real Analysis*, Reading: Addison-Wesley, 1969.

[8] R. B. Ash, *Real Analysis and Probability*, New York: Academic Press, 1972.

INDEX OF AUTHORS

Aazhang, B. 92
Adoul, J.-P. 1002
Allebach, J. P. 194
Alnuweiri, H. M. 575
Altman, E. 1147
Ando, Y. 631,912
Ansari, N. 707
Arce, G. R. 353
Ariyawansa, K. A. 483
Arvena, J. L. 595
Assal, A. F. 162
Atallah, M. J. 758
Bahr, R. K. 994
Balasubramanian, K. 296
Balakrishnan, V. 889
Bar-David, A. 868
Barbosa, V. C. 1089
Barmish, B. R. 465
Barrat, C. H. 889
Baudet, G. 585
Bhagwat, A. S. 613
Bhargava, U. K. 90
Birkemeier, W. P. 125
Boncelet,Jr., C. G. 88,383
Bougoulias, D. P. 984
Bovik, A. C. 390
Boyd, S. P. 889
Brady, M. L. 560
Brown, D. J. 560
Brown, O.M. 703
Bruckstein, A. 695,1021
Bucklew, J. A. 864,994
Cansever, D. 1149
Cantone, G. 964
Chang, T. N. 166
Chao, J. J. 974
Chau, A. 246
Chen, C. H. 214
Chen, C.-T. 665
Chow, B. S. 125
Ciciani, B. 964
Cobb, J. D. 899
Cochran, D. 901
Conrad, C. J. 485
Costello, D. J. 1037
Coyle, E. J. 363
Dandapani, R. 916
Dasgupta, S. 613
Datta, A. 643
Davison, E. J. 166,410,903

DeCarlo, R. 456
DeMarco, C. L. 465
Dean, J. A. 324
Delp, E. J. 707
Delprat, M. 1004
Derin, H. 705
Dorny, C. N. 1023
ElGindy, H. 768
Elhuni, H. 925
Eom, K.-B. 685
Ersoy, O. K. 214,709
Fam, A. T. 400
Ferrandiz, J. M. 1159
Fischer, T. R. 224,1012
Foster, M. J. 740
Freudenberg, J. S. 879
Fu, A. 787
Fuhrmann, D. 703
Fujisawa, T. 59
Gafni, E. 603,1089
Gallager, N. C. 344
Geraniotis, E. 246,540
Gesing, W. 410
Ghosh, Anjan K. 184
Gibson, J. D. 12,113,1035
Glower, J. 444
Gonzalez, T. 334,550
Goodman, D. M. 90
Goodrich, M. T. 758
Gray, S. D. 113
Gupta, A. K. 573
Halverson, D. R. 236
Hamada, K. 631,912
Hambrusch, S. E. 573
Han, J.-Y. 944
Hasham, A. 306
Hashimoto, S. 675
He, X. 777
Heitman, G. K. 623
Herro, M. A. 1047
Hester, J. H. 308
Hirschberg, D. S. 308
Hopkins, M. A. 641
Howell, R. R. 721
Hu, Y.-N. 1098
Huang, B.-C. 288
Huang, Y.-F. 115
Iftar, A. 420
Italiano, G. F. 29
Jeong, D. G. 1035
Kalyanasundaram, B. 749
Kashiwabara, T. 59
Kashyap, R. L. 90,685
Kassam, S. A. 256,363

Keller, C. 519
Kelly, P. A. 79
Kessler, I. 844
Khalil, H. 1098
Khorrami, F. 1117
Khraishi, N. M. 889
Kim, B. G. 530
Kim, C. 730
Kim, D. Y. 709
Kim, J.-H. 954
Kim, K. H. 645
Kim, Y. C. 645
King, J. T. 916
Koivo, A. K. 1108
Kumar, B. 276
Kumar, V. K. 575
Kundu, S. 935
Kurtz, D. 866
Lakshmanan, S. 705
Langston, M. A. 288
Lazar, A. A. 834, 1159
Lee, C. C. 284,974
Lee, C.-P. 485
Lee, S. C. 1132
Lee, S.-L. 550
Lee, Y. 400
Lee, Y. H. 400
Lei, T. 1023
Letaief, K. B. 286
Levy, H. 49
Li, X. 889
Lin, C. 1136
Lindenbaum, M. 695
Lindner, D. K. 160
Lingas, A. 324
Lipton, R. J. 296
Liu, Gonhsin 204
Liu, R. 175
Longbotham, H. R. 390
Longley, L. A. 284
Lucertini, M. 39
Luse, D. W. 160
Malone, K. T. 1012
Manber, U. 1064
Manohar, S. 585
Marberg, J. 603
Marcellin, M. W. 224
Marinescu, D. C. 1136
Massey, J. L. 1
Masuda, S. 59
Matthews, G. P. 456
McClamroch, N. H. 485
McGeoch, C. 19
McHugh, J. 69

Meyer, D. G. 889
Meyer, G. G. L. 473
Michel, A. N. 870
Migimatsu, T. 1132
Milito, R. A. 1149
Miller, M. I. 111,213
Mina, I. 1126
Morrison, J. M. 1169
Mukherji, U. 824
Nakamura, K. 59,1057
Narendra, K. S. 135
Nazari, N. 528
Newman-Wolfe, R. 807, 1073
Norman, S. A. 889
O'Byrne, V. 501
Oda, H. 675
Ogier, R. G. 822
Oloomi, H. 914
Oppenheimer, E. P. 870
Ossia, K. 428
Ozguner, U. 420, 444, 1117
Palmieri, F. 88
Pardalos, P. M. 812
Park, A. 296
Parker, J. 1064
Patel, P. 903
Perl, Y. 39, 69
Perry, J. E. 797
Peters, J. G. 787
Podrazik, L. J. 473
Porter, W. A. 595
Preparata, F. P. 643
Pujara, L. R. 1128
Raman, V. 175
Ramarao, K. V. 1079
Rao, A. K. 115
Reddy, S. M. 935
Reed, D. E. 511
Rego, V. 854
Rhodes, I. B. 162
Robertazzi, T. G. 834
Root, W. L. 79,623
Rosier, L. 721
Ross, K. W. 1138
Rouanne, M. 1037
Saberi, A. 150
Sack, J.-R. 306,324
Sadowsky, J. S. 286
Sain, M. K. 145
Salam, F. M. A. 655,1130
Saluja, K. K. 916
Sannuti, P. 150
Sato, Y. 675
Savage, C. D. 797

Sawan, M. E. 914
Schlegel, C. B. 1047
Schnitger, G. 749
Sciuto, D. 1134
Scott, P. D. 204
Shyamasundar, R. K. 711
Sidi, M. 844,866,868
Simeone, B. 39
Singh, R.P. 135
Sinha, A. S. C. 438
Smith, H. W. 428
Snyder, Donald L. 111,213
Snyder, L. 562
Song, I. 256,363
Soroushnejad, M. 540
Stallmann, M. 797
Stark, W. 530
Stevenson, R. L. 353
Sudborough, I. H. 730
Suzuki, M. 631,912
Szpankowski, W. 314,854
Tai, H.-M. 164
Thomas, J. B. 226,266
Thomopoulos, S. C. A. 984
Thompson, M. W. 236
Tsai, J. J. 562
Turmon, M. 111
Turner, J. 9
Tyagi, A. 562
Uppal, S. 864
VanLandingham, H. F. 641
Varadarajan, R. 1138
Varanasi, M. K. 92
Venkatesan, S. 1079
Verdu, S. 1025
Vergis, A. 925
Viswanathan, R. 276,984
Walcott, B. L. 446
Warren, D. 226
Weber, C. A. 194,344
Wickert, M. A. 511
Willett, P. K. 266
Wise, Gary 1169
Wyman, B. F. 145
Yang, H. S. 645
Yaz, E. 493,499
Yen, H.-C. 721
Zak, S. H. 446
Zheng, S.-Q. 334
Ziemer, R. E. 528

UNIVERSITY OF ILLINOIS-URBANA
621.3811AL5 C001
PROCEEDINGS / ALLERTON CONFERENCE ON COM
1986
3 0112 008149210